AF

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application. No: 09/518,492
Filed: March 3, 2000
Inventor(s):
Ram Kudukoli, Robert Dye, Melanie Jensen, and Yumiko Kawachi

Title: System and Method for Programmatically Creating a Graphical Program

§

Examiner: Vu, Kieu D.
Group/Art Unit: 2173
Atty. Dkt. No: 5150-37301

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below.

Jeffrey C. Hood

Signature 2/21/2006 Date

APPEAL BRIEF

Box: Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir/Madam:

Further to the Notice of Appeal filed October 5, 2005 and the Notice of Non-Compliant Appeal Brief mailed January 30, 2006, Appellant presents this Appeal Brief. Appellant respectfully requests that this appeal be considered by the Board of Patent Appeals and Interferences.

02/28/2006 MAHMED1 00000029 501505 09518492
01 FC:1401 500.00 DA

I. REAL PARTY IN INTEREST

The subject application is owned by National Instruments Corporation, a corporation organized and existing under and by virtue of the laws of the State of Delaware, and having its principal place of business at 11500 N. MoPac Expressway, Bldg. B, Austin, Texas 78759-3504.

II. RELATED APPEALS AND INTERFERENCES

No related appeals or interferences are known which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1-66 were originally filed in the application. In a preliminary amendment filed August 1, 2000, claims 1-66 were cancelled, and new claims 67-166 were added. In an amendment filed June 30, 2003, claims 67-166 were cancelled, and new claims 167-284 were added. In a supplemental response filed October 17, 2003, claims 167-241, 255-262, 264-181, and 284-285 were elected in response to a restriction requirement, and claims 242-254, 263, and 283 withdrawn from consideration. In an amendment filed April 13, 2004, claims 169, 206-207, and 221 were cancelled, claims 167, 170, 178-188, 190-202, 212, 214, 216-220, 222, 230-241, 255, 256, 259-261, 264, 268, 270, 272, 274, 275, 277-282, and 284 were amended. In an amendment filed October 27, 2004, claims 167-241, 255-262, 264-282, and 284 were cancelled, and new claims 285-381 were added. In an amendment filed April 4, 2005, claims 285, 286, 291, 292, 299, 302-304, 306, 307, 309, 319, 326, 331-334, 339, 340, 353, 355, 362, 363, 368-370, 372, 272, 275, and 378-381 were amended. In an amendment filed August 29, 2005, claim 355 was amended.

Claims 285-381 stand rejected under 35 U.S.C. § 103(a), and are the subject of this appeal. A copy of claims 285-381, incorporating entered amendments, as on appeal, is included in the Claims Appendix hereto.

IV. STATUS OF AMENDMENTS

An amendment to claim 355 has been filed subsequent to the final rejection in the Office Action of July 14, 2005. The Claims Appendix hereto reflects the current state of the claims.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The present application relates to the field of graphical programming, and in particular to a system and method for programmatically creating a graphical program.

Independent claim 285 recites a computer-implemented method for programmatically creating a graphical program. A first program is stored, where the first program is executable to automatically create a new graphical program based on received information, e.g., user-provided information. *See, e.g., Specification, p. 6, lines 10 – 29; p. 30, lines 1 – 22, Figures 12, 24, 27A and 27B.* The first program is executed, which automatically creates the new graphical program based on the received information, including automatically creating a plurality of graphical program nodes in the new graphical program, and automatically interconnecting the plurality of graphical program nodes in the new graphical program, where the interconnected plurality of graphical program nodes include at least a portion of the new graphical program. *See, e.g., Specification, p. 39, line 25 – p. 43, line 2; p. 30, lines 1 – 22, Figures 25, 26, 27A and 27B.* Automatically creating the new graphical program creates the new graphical program without any user input specifying the plurality of graphical program nodes or the interconnection of the plurality of graphical program nodes during the creating. *See, e.g., Specification, p. 17, lines 27-29.*

Independent claim 333 is an analogue memory medium claim to method claim 285, and is thus directed to a memory medium that stores program instructions for programmatically creating a graphical program. The program instructions are executable by a processor perform the following: A first program is stored, where the first program is executable to automatically create a new graphical program based on received information, e.g., user-provided information. *See, e.g., Specification, p. 6, lines 10 – 29; p. 30, lines 1 – 22, Figures 12, 24, 27A and 27B.* The first program is executed, which automatically creates the new graphical program based on the received information, including automatically creating a plurality of graphical program nodes in the new graphical program, and automatically interconnecting the plurality of graphical program

nodes in the new graphical program, where the interconnected plurality of graphical program nodes include at least a portion of the new graphical program. *See, e.g., Specification, p. 39, line 25 – p. 43, line 2; p. 30, lines 1 – 22, Figures 25, 26, 27A and 27B.* Automatically creating the new graphical program creates the new graphical program without any user input specifying the plurality of graphical program nodes or the interconnection of the plurality of graphical program nodes during the creating. *See, e.g., Specification, p. 17, lines 27-29.*

Independent claim 353 is directed to a memory medium that stores a first graphical program, where the first graphical program includes a graphical program creation node for automatically instantiating a new graphical program, a first object creation node for creating a first node in the new graphical program, and a second object creation node for creating a second node in the new graphical program. The first graphical program is executable to: instantiate the new graphical program (via execution of the graphical program creation node), include the first node and the second node in the new graphical program (via execution of the first and second object creation nodes), and connect the first node to the second node. *See, e.g., Specification, p. 6, lines 24 – 29; p. 21, lines 6 – 21; p. 25, lines 18 – 22; p. 30, line 1 – p. 32, line 20; p. 38, line 22 – p. 43, line 8; p. 36, lines 21 – 25, p. 42, lines 7 – 8; Figures 12, 24, 27A and 27B.* The first graphical program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes. *See, e.g., Specification, p. 17, lines 27-29.*

Independent claim 355 is directed to a computer-implemented method for automatically creating a graphical program with similar limitations as claim 353, but where the first program is not limited to a graphical program (*See, e.g., Specification, p. 23, lines 28 – 29*), and where the first program includes a graphical program creation *function* for automatically instantiating a new graphical program (as opposed to a graphical program creation *node*), as well as an object creation function for automatically including a node in the new graphical program (as opposed to an object creation *node*). *See, e.g., Specification, p. 25, lines 23 – 28; For an exemplary description of functions,*

see, p. 25, line 7 – p. 37, line 11; Figures 10-20. The first program is executed, which automatically instantiates the new graphical program, automatically includes the node in the new graphical program, and automatically connects the node to at least one other node in the new graphical program. The first program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes. *See, e.g., Specification, p. 17, line 24 – 29.*

Independent claim 362 is an analogue system claim to method claim 285, and is thus directed to a system that implements the method of that claim. More specifically, the system includes a processor and a memory medium that stores program instructions that are executable by the processor to perform the following method. *See, e.g., Specification, pp. 12 – 17; and Figures 1, 2A, 2B, and 3.* A first program is stored, where the first program is executable to automatically create a new graphical program based on received information, e.g., user-provided information. *See, e.g., Specification, p. 6, lines 10 – 29; p. 30, lines 1 – 22, Figures 12, 24, 27A and 27B.* The first program is executed, which automatically creates the new graphical program based on the received information, including automatically creating a plurality of graphical program nodes in the new graphical program, and automatically interconnecting the plurality of graphical program nodes in the new graphical program, where the interconnected plurality of graphical program nodes include at least a portion of the new graphical program. *See, e.g., Specification, p. 39, line 25 – p. 43, line 2; p. 30, lines 1 – 22, Figures 25, 26, 27A and 27B.* Automatically creating the new graphical program creates the new graphical program without any user input specifying the plurality of graphical program nodes or the interconnection of the plurality of graphical program nodes during the creating. *See, e.g., Specification, p. 17, lines 27-29.*

Independent claim 370 is directed to a system for automatically creating a graphical program with similar limitations as claim 353, but using a client/server approach. More specifically, a client program executing in a computer system performs API calls to a server program to automatically create a graphical program based on

received information. *See, e.g., Specification, p. 6, line 20 – p. 8, line 8; Figures 7, 8, and 9; For API functions, see also, p. 25, line 7 – p. 37, line 11; Figures 10-20.* The server program is executable to receive the client program calls to automatically create the graphical program based on the received information, including automatically creating a plurality of graphical program nodes in the new graphical program, and automatically interconnecting the plurality of graphical program nodes in the new graphical program, where the interconnected plurality of graphical program nodes comprise at least a portion of the new graphical program. *See, e.g., Specification, p. 21, line 20 – p. 22, line 9; Figures 7, 8, and 9.* The server program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes. *See, e.g., Specification, p. 17, lines 27-29.*

Independent claim 379 is a means plus function claim whose subject matter is a memory medium comprising a client program for automatically creating a new graphical program. According to claim 379, the client program includes:

1) a means for automatically instantiating the new graphical program, where the means for automatically instantiating the new graphical program are operable to automatically instantiate the new graphical program without any user input specifying instantiation of the new graphical program, as disclosed at least at:

p. 6, line 4, lines 10-11;

p. 13 lines 6-8;

p. 25, lines 18-21;

p. 17, line 24 – 29;

p. 30, lines 1-22; and

Figure 12;

2) a means for automatically creating a node in the new graphical program, where the means for automatically creating the node in the new graphical program are operable to automatically create the node in the new graphical program without any user input specifying creation of the node in the graphical program, as disclosed at least at:

p. 6, lines 25-27;

p. 25, lines 18 –25;

p. 31, line 23 – p. 32, line 20;

p. 17, line 24 – 29; and

Figure 14;

3) a means for getting or setting properties of the new graphical program or the node, as disclosed at least at:

p. 6, lines 25-29;

p. 35, line 18 – p. 36, line 20; and

Figure 19;

4) a means for automatically invoking methods on the new graphical program or the node, where the means for automatically invoking methods on the new graphical program or the node are operable to automatically invoke methods on the new graphical program or the node without any user input specifying invocation of methods on the new graphical program or the node, as disclosed at least at:

p. 6, lines 25-29;

p. 36, line 21 – p. 37, line 11;

p. 17, line 24 – 29; and

Figure 20.

Independent claim 381 is directed to a computer-implemented method for automatically creating a graphical program with similar limitations as claim 353, but rather than the first graphical program including first and second object creation nodes, claim 381 recites that the first graphical program includes at least one object creation node (for automatically creating at least one graphical program node in the new graphical program). The first graphical program is executed to automatically create the new graphical program, including automatically instantiating the new graphical program, and automatically including the node in the new graphical program. *See, e.g., Specification, p. 38, line 22 – p. 43, line 2; p. 30, lines 1 – 22, Figures 25, 26, 27A and 27B.* As

described above, the first graphical program automatically creates the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes. Additionally, the first graphical program automatically includes the node in the new graphical program without any user input specifying inclusion of the node in the new graphical program. *See, e.g., Specification, p. 17, lines 27-29.*

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 370-380 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Sojoodi et al (U.S. Patent No. 6,437,805, "Sojoodi") and Yoshida et al (U.S. Patent No. 5,767853, "Yoshida").

Claims 285-369 and 381 stand rejected under 35 U.S.C. 103(a) as being unpatentable over McDonald et al (USP 5,6966,532,."McDonald"), Sojoodi, and Yoshida.

VII. ARGUMENT

Section 103(a) Rejections

Claims 370-380 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Sojoodi et al (U.S. Patent No. 6,437,805, "Sojoodi") and Yoshida et al (U.S. Patent No. 5,767853, "Yoshida"). Appellant respectfully traverses this rejection for the following reasons. Different groups of claims are addressed under their respective subheadings.

Claim 370

Claim 370 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

For example, Appellant notes that independent claim 370 includes the limitation that the new graphical program is created *automatically*, i.e., *without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes.*

The Examiner asserts that "Sojoodi teaches a system for automatically creating a graphical program..."; however, Sojoodi describes *manual* creation of a graphical program which is operable to access capabilities of an object, and specifically does *not* describe automatically creating the graphical program via execution of a program. Appellant notes that the claims as currently written emphasize the fact that the graphical program is *not* generated manually, i.e., is not generated in response to user input specifying selection of graphical program nodes and interconnection of graphical program nodes, but rather is generated *automatically*, i.e., without any user input specifying the plurality of graphical program nodes or the interconnection of the plurality of graphical program nodes during said creating. Appellant notes that the Background section, page 5, lines 15-22, recites:

> As described above, a user typically creates a graphical program within a graphical programming environment by ***interactively or manually placing icons or nodes representing the desired blocks of functionality on a diagram, and connecting the icons/nodes together*** to represent the data flow of the program. The ability to ***programmatically create/edit graphical programs*** enables a graphical program to ***automatically*** be created/edited ***without this type of interactive user intervention.*** A

> system and method for programmatically creating/editing graphical programs is described herein. (*emphasis added*)

It is this type of graphical program creation that is disclosed by Sojoodi. In direct contrast, the Summary, page 6, lines 4-9 recites:

> The ability to programmatically create/edit graphical programs enables applications and tools to ***automatically*** create or modify a graphical program or a portion of a graphical program. In the preferred embodiment, any operation which a user may perform while interactively assembling or editing a graphical program (e.g., connect two objects, change the position of an object, change the color of an object, etc.) may be performed programmatically. (*emphasis added*)

Similarly, regarding client/server embodiments, page 8, lines 1-4 recites:

> As noted above, the program which calls the API in order to ***automatically*** create or edit a graphical program is referred to herein as the client program. The program which performs the create/edit operations on the graphical program is referred to herein as the server program. (*emphasis added*)

The Examiner has repeatedly asserted that Sojoodi discloses a client program (executing in the computer system) performing API calls to programmatically create a graphical program, citing col. 5, lines 28-30. However, the cited passage actually reads:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

Nowhere does the cited portion (or any other portion of Sojoodi) refer to "automatically" creating a graphical program. Rather, Sojoodi discloses (Abstract):

> During creation of the graphical program, ***the user*** operates to place an object node in the graphical program, wherein the object node is operable to access capabilities of the object. This preferably includes ***the user*** arranging on the screen the graphical program, including the object node and various other nodes, and connecting the various nodes to create the graphical program. (*emphasis added)*

Clearly, in Sojoodi's system, the *user* selects, arranges, and connects the nodes. Similarly, the Examiner cites col. 4, lines 43-53, in asserting that Sojoodi discloses a server program operable to receive the client program calls to programmatically create a

graphical program and operable to perform the respective operations. The cited portion actually reads:

> The notion of object technology relates to an application program or object using capabilities or services of an object. For example, object technology includes the notion of "servers" "exporting" their "objects" for use by "clients." A server is a software application which makes available its classes to be accessed from another application, referred to as a client. A client instantiates objects from the classes in the type libraries. In addition, the term "object" also includes various other types of components or objects, including applications, which offer services or functionality that can be accessed by a client.

Nowhere does the cited portion (or any other portion) refer to a server program operable to receive client program calls to *automatically* create a graphical program based on received information and operable to perform the respective operations, i.e., to automatically create the graphical program.

The Final Office Action states that "Sojoodi does not teach creating the new program from a program without any user input specifying the nodes and interconnection of the plurality of node" [sic]. Appellant submits that this statement is itself misleading, in that the particular wording "creating the new program from a program" implies that a new program is necessarily created *based on* another program, i.e., that the new program is necessarily created by converting or translating another program, which is *not* an accurate description or characterization of Appellant's invention as claimed.

Thus, Appellant respectfully submits that Sojoodi does not disclose automatically creating the graphical program without user input specifying or guiding the inclusion and interconnection of graphical program nodes, and more specifically fails to teach a client program performing API calls to automatically create a graphical program, and in response, a server program automatically creating a plurality of graphical program nodes in the new graphical program, and automatically interconnecting the plurality of graphical program nodes in the new graphical program.

The Office Action asserts that Yoshida corrects the deficiencies of Sojoodi. Appellant respectfully disagrees. Yoshida is directed to invocation of an application in response to user-designation of a data file based on past user operations, e.g., based on

the frequency of past associations between the application and the data file, based on the last application that used the data file, etc. Nowhere does Yoshida teach or suggest automatically creating a graphical program. More specifically, Yoshida fails to teach automatic creation of a graphical program without any user input specifying the nodes and interconnection of the plurality of node, nor a server program executable to perform such automatic creation the graphical program. Appellant respectfully submits that Yoshida's system and method for creating a macro via a macro-creating program without user input is not germane to automatic creation of a graphical program, and in no way discloses means for doing so. Appellant notes that Yoshida never mentions automatic creation of a graphical program in response to received information, nor does Yoshida even mention a graphical program at all.

In other words, as argued above, neither Sojoodi nor Yoshida teaches or suggests automatic creation of a graphical program by *any* means. Thus, for at least the reasons provide above, Appellant submits that neither Sojoodi nor Yoshida, taken singly or in combination, teaches or suggests all the features and limitations of claim 370 and so claim 370, and those claims dependent therefrom, are patentably distinct and non-obvious over Sojoodi and Yoshida, and are thus allowable.

Claims 372 and 373

In addition to the distinctions noted above in regard to claim 370, the cited art fails to teach "wherein the client program performs said calls to automatically create a graphical program by obtaining a reference to a software component and invoking methods of the software component; and wherein the software component is operable to perform the operations of automatically creating the graphical program", as recited in claim 372. Nor does the cited art teach or suggest "wherein the client program performs said calls to automatically create a graphical program by obtaining a reference to a software component and invoking methods of the software component; and wherein the software component relays the client program calls to the server program", as recited in claim 373.

The Examiner asserts that Sojoodi teaches "the creating a graphical program by obtaining a reference to a software component", citing col. 58, lines 42-44, and "invoking

methods of the software component", citing col. 6, lines 15-19. However, col. 58, lines 42-44 recite:

> ...wherein the pre-existing software object was constructed in response to text-based programming language source code.

Nowhere does the cited portion (or any other portion) refer to a client program performing calls to *automatically create a graphical program* by obtaining a reference to a software component.

Col. 6, lines 15-19 recite:

> Where the object is a software object or component, the object node is executable to either invoke a method of the object, get and/or set one or more properties of the object, or access other capabilities of the object.

Clearly, the cited text does not disclose invoking methods of the software component *to automatically create a graphical program*. In fact, as noted above, Sojoodi (and Yoshida) fails to disclose automatic generation of a graphical program at all, and so cannot possibly teach these features of claims 372 and 373.

Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claims 272 and 273, and so claims 272 and 273 are allowable.

Claim 374

In addition to the distinctions noted above in regard to claim 370, the cited art fails to teach "wherein the server program is a graphical programming environment application", as recited in claim 374.

In asserting that Sojoodi teaches this feature, the Examiner cites Figure 1A. However, Figure 1A discloses illustrates an industrial automation system, including a computer which connects to one or more devices or instruments. The computer comprises a CPU, a display screen, memory, and one or more input devices such as a mouse or keyboard as shown. The computer connects through the one or more devices to a process or device to perform an automation function, such as MMI (Man Machine Interface), SCADA (Supervisory Control and Data Acquisition), portable or distributed data acquisition, process control, advanced analysis, or other control.

Nowhere in Figure 1A or the related text (or anywhere else) does Sojoodi disclose a server program for automatically creating a graphical program itself being a graphical programming environment application.

Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 274, and so claim 274 is allowable.

Claim 375

In addition to the distinctions noted above in regard to claim 370, the cited art fails to teach "wherein the client program is a client graphical program; wherein the client graphical program includes a graphical program creation node for automatically instantiating a new graphical program; wherein the client graphical program also includes an object creation node for automatically creating a graphical program node in the new graphical program; and wherein said API calls to automatically create a graphical program comprise calls resulting from executing the graphical program creation node and the object creation node.", as recited in claim 375.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 27-30, which recites:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

Clearly, the cited text does not disclose these features of claim 375, since it makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, the client graphical program including a graphical program creation node for automatically instantiating a new graphical program, and an object creation node for automatically creating a graphical program node in the new graphical program, and a client graphical program making API calls to automatically create a graphical program, where the calls result from the execution of the graphical program creation node and the object creation node. Appellant further submits that Yoshida nowhere discloses such automatic creation of a graphical program, nor does Yoshida even mention graphical programs at all, and particularly fails to disclose a

graphical program creation node for *automatically* instantiating a new graphical program; wherein the client graphical program also includes an object creation node for *automatically* creating a graphical program node in the new graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 275, and so claim 275 is allowable.

Claim 376

In addition to the distinctions noted above in regard to claim 370, the cited art fails to teach "wherein the client graphical program further includes a property node for getting or setting a property of the graphical program node", as recited in claim 376, where the property node is used by the client graphical program to *automatically* create or edit a graphical program.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 48-52, which recites:

> The user then configures the object node to receive information on the object, preferably by the user configuring the object node with a reference to the object, e.g., a pointer, address, or other information which specifies the identity and/or location of the object...

and col. 6, lines 15-19, which recites:

> Where the object is a software object or component, the object node is executable to either invoke a method of the object, get and/or set one or more properties of the object, or access other capabilities of the object.

Nowhere does the cited text disclose these features of claim 376. While Sojoodi does disclose a property node for accessing (setting/getting) properties of an object, Appellant notes that nowhere does Sojoodi (or Yoshida) mention or even hint at using such a node in a client graphical program to automatically create a graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 376, and so claim 376 is allowable.

Claim 377

In addition to the distinctions noted above in regard to claim 370, the cited art fails to teach "wherein the client graphical program further includes an invoke node for invoking a method on the graphical program node", as recited in claim 377, where the invoke node is used by the client graphical program to *automatically* create or edit a graphical program.

In asserting that Sojoodi teaches these features, the Examiner cites col. 6, lines 15-19, which recites:

> Where the object is a software object or component, the object node is executable to either invoke a method of the object, get and/or set one or more properties of the object, or access other capabilities of the object.

Nowhere does the cited text disclose these features of claim 377. While Sojoodi does disclose an object node for invoking a method of the object, Appellant notes that nowhere does Sojoodi (or Yoshida) mention or even hint at using such a node in a client graphical program to automatically create a graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 377, and so claim 377 is allowable.

Claim 378

In addition to the distinctions noted above in regard to claim 370, the cited art fails to teach "wherein the object creation node is a first object creation node for automatically creating a first graphical program node in the new graphical program; wherein the graphical program also includes a second object creation node for automatically creating a second graphical program node in the new graphical program; and wherein the invoked method connects the first graphical program node to the second graphical program node", as recited in claim 378.

In asserting that Sojoodi teaches these features, the Examiner cites col. 57, lines 34-42, which recites:

> The memory medium of claim 57, wherein the graphical program comprises a plurality of nodes, wherein the graphical program mdevelopment environment further comprises program instructions executable to: arrange on the display screen said plurality of nodes,

including said first node; and connect said plurality of nodes and said first node ***in response to user input*** to create the graphical program.

Clearly, the cited text does not disclose these features of claim 378, since the arranging and connecting of the nodes is performed *in response to user input*, and is thus not performed *automatically*. Appellant notes that nowhere does Sojoodi (or Yoshida) mention or even hint at a second (or first) object creation node for automatically creating a second graphical program node in the new graphical program. Nor does Sojoodi (or Yoshida) mention or describe that the invoked method connects the first graphical program node to the second graphical program node, i.e., automatically, i.e., without any user input specifying the connection (or creation of the node).

Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 378, and so claim 378 is allowable.

Claim 379

Claim 379 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

For example, nowhere do Sojoodi and Yoshida disclose a client program for *automatically creating a new graphical program* including "a means for automatically instantiating the new graphical program, wherein said means for automatically instantiating the new graphical program are operable to automatically instantiate the new graphical program without any user input specifying instantiation of the new graphical program". Nor do Sojoodi and Yoshida disclose "a means for automatically creating a node in the new graphical program, wherein said means for automatically creating the node in the new graphical program are operable to automatically create the node in the new graphical program without any user input specifying creation of the node in the graphical program", nor "a means for getting or setting properties of the new graphical program or the node", nor "a means for automatically invoking methods on the new graphical program or the node, wherein said means for automatically invoking methods on the new graphical program or the node are operable to automatically invoke methods

on the new graphical program or the node without any user input specifying invocation of methods on the new graphical program or the node", as recited in claim 379.

The Examiner asserts that Sojoodi teaches manual versions of these limitations, then asserts that Yoshida provides the automatic aspect of Appellant's invention as represented in claim 379. However, as argued above, Yoshida nowhere discloses or even hints at any means for automatic creation of a graphical program, automatic setting/getting of properties of the new graphical program or node, automatic invocation of methods on the new graphical program or node, nor mentions or hints at graphical programs at all, and so does not and cannot provide these admitted missing features and limitations of claim 379. As stated earlier, Yoshida's macro-generating program for automatically creating macros in no way discloses automatic generation of a graphical program, nor is Yoshida's macro-generation technique applicable to graphical programs. In fact, because Yoshida's system and method are specifically addressed to macros, which are quite different from graphical programs, Yoshida actually teaches away from Appellant's invention as claimed.

Thus, Appellant submits that nowhere do Sojoodi and Yoshida, taken singly or in combination, teach or suggest automatic creation of a graphical program at all, and more specifically fail to teach or suggest the subject matter of claim 379, and so claim 379 and those claims dependent therefrom are allowable.

Appellant notes that claim 379, being a means plus function claim, may be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof. Appellant refers the Examiner to the various citations from the Specification presented in the above summary for independent claim 379.

Claim 380

In addition to the distinctions noted above in regard to claim 379, the cited art fails to teach "wherein the client program is a graphical program; wherein said means for automatically instantiating the new graphical program comprises a graphical program creation node; wherein said means for automatically creating a node in the new graphical program comprises an object creation node; wherein said means for getting or setting

properties of the new graphical program or the node comprises a property node; and wherein said means for automatically invoking methods on the new graphical program or the node comprises an invoke node", as recited in claim 380.

As argued above with respect to claims 375, 376, and 377, neither Sojoodi nor Yoshida discloses automatic generation of a graphical program at all, and specifically fail to disclose a graphical client program including (and executing) these respective nodes to automatically generate the new graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 380, and so claim 380 is allowable.

Appellant further submits that neither Sojoodi nor Yoshida provide a proper motivation to combine, noting that the Examiner's suggested motivation, "so that a new program can be automatically generated from an existing program without requiring user to possess a specialized knowledge about computer language" is simply a statement of presumed benefits of the alleged combination, which is simply hindsight analysis, and improper. Nowhere does Sojoodi or Yoshida indicate the desirability of a first program executable to automatically create a graphical program. Thus, Appellant respectfully submits that the Examiner's attempted combination of Sojoodi and Yoshida is improper. Moreover, Appellant submits that even were Sojoodi and Yoshida properly combinable, which Appellant argues they are not, the resulting combination would still not produce Appellant's invention as claimed, as argued at length above.

Claims 285-369 and 381 stand rejected under 35 U.S.C. 103(a) as being unpatentable over McDonald et al (USP 5,6966,532, "McDonald"), Sojoodi, and Yoshida. Appellant respectfully traverses this rejection for the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 285, 291, 314, 315, 316, 317, 318, 333, 339, 362

Claim 285 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

The Examiner admits that McDonald fails to teach "the creating a graphical program by creating graphical program nodes [sic]", but asserts that this feature is taught by Sojoodi. The Examiner further admits that neither McDonald nor Sojoodi discloses "creating the new program from a program without any user input specifying the nodes and interconnection of the plurality of node [sic]", but then asserts that Yoshida teaches this feature. Appellant respectfully disagrees, submits that the cited art, taken singly or in combination, fails to teach all the features and limitations of claim 285.

Appellant notes that McDonald is directed to generating a graphical program (or portion) by programmatically selecting *pre-existing code templates* based on user-selected graphical user interface controls and inserting the code templates into a graphical program, which is not at all the same as the present invention as represented in claim 285. The Examiner asserts that McDonald teaches "the creating of a first program, when executing, programmatically creating a new graphical program" [sic], citing col. 3, lines 61-63.

However, Appellant notes that the cited portion of McDonald actually reads:

> The present invention comprises a computer-implemented system and method for automatically generating graphical code in a graphical programming system.

Appellant submits that there are numerous possible ways to automatically generate graphical code in a graphical programming system, and that the method disclosed by McDonald is not at all the same as that claimed in the present application. Nowhere does McDonald teach or suggest "storing a first program, wherein the first program is executable to automatically create a new graphical program", nor executing the first program to automatically create the new graphical program, *including automatically creating a plurality of graphical program nodes in the new graphical program, and automatically interconnecting the plurality of graphical program nodes in the new graphical program*...wherein said automatically creating the new graphical program creates the new graphical program without any user input specifying the

plurality of graphical program nodes or the interconnection of the plurality of graphical program nodes program during said creating.

The graphical code templates selected in McDonald have preconfigured nodes and interconnections, and hence McDonald does not perform, and has no reason to perform, any automatic creation or interconnection of nodes. Nor does Sojoodi disclose automatic creation of a graphical program, nor automatic generation of nodes and automatic interconnection thereof, as mentioned above. As also mentioned above, while Yoshida does describe automatic creation of macros, nowhere does Yoshida disclose *automatically creating a plurality of graphical program nodes in the new graphical program, and automatically interconnecting the plurality of graphical program nodes in the new graphical program*, and, in fact, fails to mention or even hint at graphical programs at all.

Appellant further respectfully submits that neither McDonald, Sojoodi, nor Yoshida provides a motivation to combine, and so the attempted combination of Sojoodi and Yoshida, and of McDonald, Sojoodi, and Yoshida is improper. For example, nowhere does Sojoodi or Yoshida suggest the desirability or benefit of automatically creating a graphical program, i.e., without user input selecting the nodes and connections between the nodes. Appellant notes that the only motivation to combine indicated by the Examiner is "to automatically translate graphical program from one programming language to another programming language [sic]", which Appellant submits is not a feature of Appellant's invention as claimed, and so is an improper motivation to combine. Similarly, nowhere does McDonald indicate the desirability or benefit of automatically creating a graphical program in the manner claimed, e.g., by automatically creating and connecting one or more nodes without user input specifying the nodes or connections.

Thus, for at least these reasons, Appellant submits that McDonald, Sojoodi, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 285, and so claim 285 is allowable.

Claim 286, 334

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the new graphical program comprises a diagram portion

comprising the plurality of interconnected nodes and a user interface portion; and wherein said automatically creating the new graphical program comprises creating the diagram portion and the user interface portion."

Appellant respectfully submits that while McDonald and Sojoodi disclose graphical programs with a diagram portion and a user interface portion, neither reference teaches *wherein said automatically creating the new graphical program comprises creating the diagram portion and the user interface portion*, as claimed, since, as argued at length above, Sojoodi fails to disclose automatic creation of graphical programs at all, and McDonald's approach relies on direct user input specifying the creation, and further, relies on pre-existing graphical code templates, and specifically does not include *automatically creating a plurality of graphical program nodes in the new graphical program; and automatically interconnecting the plurality of graphical program nodes in the new graphical program*, as claimed.

For example, in the system and method of McDonald, a wizard selects a graphical code template *in response to user selection of a user interface control*, and includes the template in a graphical program, where the graphical code template corresponds to the selected control, and operates in conjunction with the control. In other words, the wizard selects a graphical code template in response to (and corresponding to) a user-selected user interface control, and inserts the template into the graphical program. Thus, a pre-existing graphical code template is selected based on a user-selected user interface control, and inserted into a graphical program, which actually teaches away from Appellant's invention as claimed. As argued above, Yoshida fails to remedy these deficiencies of McDonald and Sojoodi, and so Appellant respectfully submits that the cited art of McDonald, Sojoodi, and Yoshida, fail to disclose all the features and limitations of claim 286, and so claim 286 is allowable for at least the reasons provided above.

Claim 287, 335

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the new graphical program comprises a data flow diagram". Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken

singly or in combination, teach or suggest automatic creation of a data flow diagram in the manner claimed, and so for at least this reason, claim 287 is allowable.

Claim 288, 336

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the new graphical program comprises a virtual instrument". Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest automatic creation of a virtual instrument in the manner claimed, and so for at least this reason, claim 287 is allowable.

Claim 289, 337

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein said executing the first program occurs in a first computing environment; wherein said first computing environment is connected to a second computing environment; wherein said executing the first program comprises sending information from the first computing environment to the second computing environment; and wherein the new graphical program is created in the second computing environment".

In asserting that McDonald teaches this feature, the Examiner cites step 202 of Figure 2, and col. 11, lines 18-19. Appellant respectfully disagrees.

Step 202 of Figure 2 states "Select a control which represents input to or output from a graphical program in response to user input". Col. 11, lines 18-19 read:

> ...selection of a control from this special palette in step 202 automatically initiates the graphical code generation wizard in step 204.

Nowhere do the cited Figure and text mention or even hint at such use of multiple computing environments in the automatic generation of graphical programs as recited in claim 289. Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 289, and so for at least these reasons, claim 289 is allowable.

Claim 290, 338

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach " wherein the first program specifies creation of the new graphical program".

The Examiner asserts that McDonald teaches "the specifying the creation of the new program" citing col. 3, lines 61-66. Appellant respectfully disagrees, and notes that the Examiner has omitted the detail that the *first program* specifies the creation.

Col. 3, lines 61-66 read:

> The present invention comprises a computer-implemented system and method for automatically generating graphical code in a graphical programming system. The computer memory stores a plurality of graphical code portions, also referred to as graphical code templates or G code templates.

Nowhere does the cited text mention or even hint at *a first program* specifying the automatic creation of the new graphical program as claimed. Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 290, and so for at least these reasons, claim 290 is allowable.

Claim 292, 340

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach " wherein the first program comprises a client program that calls an application programming interface (API) to automatically create the new graphical program; and wherein said executing comprises the first program calling the API to perform said creating and said interconnecting".

The Examiner asserts that Sojoodi teaches "wherein the first program comprises a client program that calls an application programming interface (API) to automatically create the new graphical program; and wherein said executing comprises the first program calling the API to perform said creating and said interconnecting", citing col. 5, lines 28-30. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

Nowhere does the cited portion (or any other portion of Sojoodi) refer to "automatically" creating a graphical program. Rather, as noted above, Sojoodi discloses (Abstract):

> During creation of the graphical program, ***the user*** operates to place an object node in the graphical program, wherein the object node is operable to access capabilities of the object. This preferably includes ***the user*** arranging on the screen the graphical program, including the object node and various other nodes, and connecting the various nodes to create the graphical program. *(emphasis added)*

As argued at length above, in Sojoodi's system, the *user* selects, arranges, and connects the nodes. Similarly, the Examiner cites col. 4, lines 43-53, in asserting that Sojoodi discloses a server program operable to receive the client program calls to programmatically create a graphical program and operable to perform the respective operations. However, the cited portion actually reads:

> The notion of object technology relates to an application program or object using capabilities or services of an object. For example, object technology includes the notion of "servers" "exporting" their "objects" for use by "clients." A server is a software application which makes available its classes to be accessed from another application, referred to as a client. A client instantiates objects from the classes in the type libraries. In addition, the term "object" also includes various other types of components or objects, including applications, which offer services or functionality that can be accessed by a client.

Nowhere does the cited portion (or any other portion) refer to a server program operable to receive client program calls to *automatically* create a graphical program based on received information and operable to perform the respective operations, i.e., to automatically create the graphical program.

Appellant thus respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 292, and so for at least these reasons, claim 292 is allowable.

Claim 293, 341

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the first program comprises a server program; and wherein said executing comprises the first program receiving calls from a client program".

In asserting that Sojoodi teaches this feature, the Examiner cites Figure 1A. However, Figure 1A discloses illustrates an industrial automation system, including a computer which connects to one or more devices or instruments. The computer comprises a CPU, a display screen, memory, and one or more input devices such as a mouse or keyboard as shown. The computer connects through the one or more devices to a process or device to perform an automation function, such as MMI (Man Machine Interface), SCADA (Supervisory Control and Data Acquisition), portable or distributed data acquisition, process control, advanced analysis, or other control.

Nowhere in Figure 1A or the related text (or anywhere else) does Sojoodi disclose a server program for automatically creating a graphical program, nor executing the server program including receiving calls from a client program to automatically create the new graphical program. Thus, for at least these reasons, Appellant submits that Sojoodi and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 293, and so claim 293 is allowable.

Claim 294, 342

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the new graphical program comprises a block diagram, and wherein the plurality of graphical program nodes comprise at least one function node placed in the block diagram".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 16, lines 15-17. Appellant respectfully disagrees, noting that col. 16, lines 15-17 read:

> In another embodiment, the user assembles user interface nodes into the block diagram with the function nodes...

Nowhere does the cited text mention or even hint at including at least one function node placed in the block diagram as part of automatic creation of a graphical program, as claimed. Appellant respectfully submits that nowhere do McDonald,

Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 294, and so for at least these reasons, claim 294 is allowable.

Claim 295, 343

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the new graphical program includes a block diagram, and wherein the plurality of graphical program nodes comprise at least one programmatic structure placed in the block diagram".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 16, lines 15-17. Appellant respectfully disagrees, noting that col. 16, lines 15-17 read:

> In another embodiment, the user assembles user interface nodes into the block diagram with the function nodes...

Nowhere does the cited text mention or even hint at including at least one programmatic structure placed in the block diagram as part of automatic creation of a graphical program, as claimed. Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 295, and so for at least these reasons, claim 295 is allowable.

Claim 296, 344

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the plurality of graphical program nodes comprise at least one graphical loop structure".

In asserting that McDonald teaches this feature, the Examiner cites col. 16, lines 25-27. Appellant respectfully disagrees, noting that col. 16, lines 25-27 read:

> In the preferred embodiment, the locking mechanism is provided only for code encapsulated in a structure node...

Nowhere does the cited text mention or even hint at including at least one graphical loop structure created as part of automatic creation of a graphical program, as claimed. Appellant respectfully submits that nowhere do McDonald, Sojoodi, and

Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 296, and so for at least these reasons, claim 296 is allowable.

Claim 297, 345

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the plurality of graphical program nodes comprise at least one graphical case structure".

In asserting that McDonald teaches this feature, the Examiner cites col. 16, lines 25-27. Appellant respectfully disagrees, noting that col. 16, lines 25-27 read:

> In the preferred embodiment, the locking mechanism is provided only for code encapsulated in a structure node...

Nowhere does the cited text mention or even hint at including at least one graphical case structure created as part of automatic creation of a graphical program, as claimed. Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 297, and so for at least these reasons, claim 297 is allowable.

Claim 298, 346

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein said interconnecting the plurality of graphical program nodes comprises displaying a connection between an input of a first graphical program node and an output of a second graphical program node".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 15, lines 26-31. Appellant respectfully disagrees, noting that col. 15, lines 26-31 read:

> In the preferred embodiment, the nodes are connected in a data flow paradigm, wherein nodes include inputs and outputs, and each node receives input data from another node, or from an input user interface node or terminal, and each node provides output data to another node, or to an output user interface node or terminal.

Nowhere does the cited text disclose displaying a connection between an input of a first graphical program node and an output of a second graphical program node as part of automatically interconnecting the plurality of (automatically created) graphical

program nodes in the new graphical program, as claimed. Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 298, and so for at least these reasons, claim 298 is allowable.

Claim 299, 347

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "automatically creating one or more user interface nodes, wherein the one or more user interface nodes perform one or more of providing input to or displaying output from the new graphical program".

In asserting that McDonald teaches this feature, the Examiner cites col. 4, lines 3-7. Appellant respectfully disagrees, noting that col. 4, lines 3-7 read:

> The graphical programming system executing on the computer system also includes a plurality of front panel objects or controls which represent or indicate input to or output from a graphical program, or represent devices in the graphical program.

Nowhere does the cited text mention or even hint at automatic creation of user interface nodes as part of automatically generating a graphical program, as claimed. Rather, as noted above, McDonald retrieves pre-existing code templates based on user selected user interface controls. Nowhere does McDonald disclose automatically creating such nodes.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 299, and so for at least these reasons, claim 299 is allowable.

Claim 300, 348

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the new graphical program includes a user interface panel, and wherein plurality of graphical program nodes comprise at least one user interface node placed in the user interface panel".

In asserting that McDonald teaches this feature, the Examiner cites col. 4, lines 3-7. Appellant respectfully disagrees, noting that col. 4, lines 3-7 read:

> The graphical programming system executing on the computer system also includes a plurality of front panel objects or controls which represent or indicate input to or output from a graphical program, or represent devices in the graphical program.

Nowhere does the cited text mention or even hint at automatic creation of at least one user interface node placed in a user interface panel as part of automatically generating a graphical program, as claimed. Rather, as noted above, McDonald retrieves pre-existing code templates based on user selected user interface controls. Nowhere does McDonald disclose automatically creating such nodes.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 300, and so for at least these reasons, claim 300 is allowable.

Claim 301, 349

In addition to the distinctions noted above in regard to claims 285 and 300, the cited art fails to teach "wherein the user interface node comprises at least one of:

a user interface input node placed in the user interface panel for providing user input to the new graphical program; and/or

a user interface output node placed in the user interface panel for viewing output of the new graphical program".

In asserting that McDonald teaches this feature, the Examiner cites col. 4, lines 3-7. Appellant respectfully disagrees, noting that col. 4, lines 3-7 read:

> The graphical programming system executing on the computer system also includes a plurality of front panel objects or controls which represent or indicate input to or output from a graphical program, or represent devices in the graphical program.

Nowhere does the cited text mention or even hint at automatic creation of a user interface input node and/or a user interface output node placed in a user interface panel as part of automatically generating a graphical program, as claimed. Rather, as noted above, McDonald retrieves pre-existing code templates based on user selected user interface controls. Nowhere does McDonald disclose automatically creating such nodes.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 301, and so for at least these reasons, claim 301 is allowable.

Claim 302, 350

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein said automatically creating the new graphical program further comprises: obtaining a reference to a graphical program node, wherein the reference is used to manipulate the graphical program node".

In asserting that McDonald teaches this feature, the Examiner cites col. 19, lines 4-6. Appellant respectfully disagrees, noting that the cited text reads:

> In the preferred embodiment, the controls in FIGS. 7 and 8 are referenced by the graphical programming system, e.g., LabVIEW, by name.

Nowhere does the cited text mention or even hint at obtaining a reference to a graphical program node, where the reference is used to manipulate the graphical program node as part of automatically generating a graphical program, as claimed. Rather, as noted above, McDonald retrieves pre-existing code templates based on user selected user interface controls. Nowhere does McDonald disclose obtaining a reference to a graphical program node for such purposes.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 302, and so for at least these reasons, claim 302 is allowable.

Claim 303, 351

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein said automatically creating the new graphical program further comprises: performing at least one of getting or setting a property of a graphical program node".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, lines 48-52, which recites:

> The user then configures the object node to receive information on the object, preferably by the user configuring the object node with a reference to the object, e.g., a pointer, address, or other information which specifies the identity and/or location of the object...

and col. 5, lines 66-67, which recites:

> Creation of the graphical program may also include configuring a user interface to display data input to and/or output from the graphical program.

Nowhere does the cited text mention or even hint at getting or setting a property of a graphical program node as part of automatically generating a graphical program, as claimed. Rather, as noted above, Sojoodi discloses manual creation of the graphical program, as well as manually setting/getting properties of graphical program nodes. Nowhere does Sojoodi disclose automatically generating a graphical program at all, and specifically fails to disclose getting/setting properties of a node for such purposes.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 303, and so for at least these reasons, claim 303 is allowable.

Claim 304, 352

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein said automatically creating the new graphical program further comprises: invoking a method on a graphical program node".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, lines 53-65, which recites:

> In the preferred embodiment, the object node includes an object reference input for receiving a reference to the object, and the user connects the object reference input of the object node to receive the reference to the object. This preferably includes the user placing an object reference node in the graphical program, wherein the object reference node includes an object reference output that provides the reference to the object, and the user connecting the object reference output of the object reference node to the object reference input of the object node. The object node then receives the information on the object on the object reference input during execution of the graphical program.

Nowhere does the cited text mention or even hint at invoking a method on a graphical program node as part of automatically generating a graphical program, as claimed. Rather, as noted above, Sojoodi discloses manual creation of the graphical program. Nowhere does Sojoodi disclose automatically generating a graphical program at all, and specifically fails to disclose invoking a method on a graphical program node to perform such automatic creation.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 304, and so for at least these reasons, claim 304 is allowable.

Claim 305, 367

In addition to the distinctions noted above in regard to claim 285, the cited art fails to teach "wherein the first program is a first graphical program".

The Examiner admits that McDonald fails to teach this feature, but then asserts that Sojoodi remedies this deficiency, citing col. 5, lines 28-32. Appellant respectfully disagrees, noting that the cited text reads:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

Nowhere does the cited portion (or any other portion of Sojoodi) refer to "automatically" creating a graphical program, specifically via execution of a graphical program, as claimed. Thus, Sojoodi does not, and cannot, disclose a first graphical program executable to automatically create a new graphical program, as claimed. Moreover, as noted above, Sojoodi fails to indicate the desirability of such automatic creation of graphical programs. Nor does Yoshida suggest the desirability of a graphical program that executes to automatically create a new graphical program. Thus, Appellant submits that the attempted combination of McDonald and Sojoodi (and Yoshida) is improper.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 305, and so for at least these reasons, claim 305 is allowable.

Claims 306, 368

In addition to the distinctions noted above in regard to claims 285 and 305, the cited art fails to teach "wherein the first graphical program includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program; and wherein said automatically creating the new graphical program comprises including the at least one graphical program node in the new graphical program".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, lines 27-30. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

Clearly, the cited text does not disclose such an object creation node, nor a first graphical program executable to automatically create a new graphical program, as claims. Moreover, nowhere does the cited portion (or any other portion of Sojoodi) refer to "automatically" creating a graphical program at all. Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program.

Appellant thus respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 306, and so for at least these reasons, claim 306 is allowable.

Claim 307

In addition to the distinctions noted above in regard to claims 285, 305, and 306, the cited art fails to teach "wherein the first graphical program further includes a property node; and wherein said automatically creating the new graphical program comprises the property node getting or setting a property of the graphical program node".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, lines 48-52, which recites:

> The user then configures the object node to receive information on the object, preferably by the user configuring the object node with a reference

> to the object, e.g., a pointer, address, or other information which specifies the identity and/or location of the object...

and col. 5, lines 66-67, which recites:

> Creation of the graphical program may also include configuring a user interface to display data input to and/or output from the graphical program.

Nowhere does the cited text mention or even hint at a first graphical program (executable to automatically create a new graphical program) including a property node, and where automatically creating the new graphical program includes the property node getting or setting a property of a (automatically generated) graphical program node, as claimed. Rather, as noted above, Sojoodi discloses manual creation of the graphical program, as well as manually setting/getting properties of graphical program nodes. Nowhere does Sojoodi disclose automatically creating a graphical program at all, and specifically fails to disclose using a property node for such purposes.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 307, and so for at least these reasons, claim 307 is allowable.

Claim 308

In addition to the distinctions noted above in regard to claims 285, 305, 306, and 307, the cited art fails to teach "wherein the object creation node outputs a reference to the graphical program node; wherein the property node receives as input the reference to the graphical program node; and wherein the property node gets or sets a property of the graphical program node specified by the reference to the graphical program node".

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, lines 48-52, which recites:

> The user then configures the object node to receive information on the object, preferably by the user configuring the object node with a reference to the object, e.g., a pointer, address, or other information which specifies the identity and/or location of the object...

and col. 5, lines 66-67, which recites:

> Creation of the graphical program may also include configuring a user interface to display data input to and/or output from the graphical program.

As argued above with reference to claims 305, 306 and 307, nowhere does the cited text mention or even hint at a first graphical program (executable to automatically create a new graphical program) including a property node or an object creation node. Moreover, the cited text also fails to disclose the object creation node outputting a reference to the (automatically generated) graphical program node, which is received as input by the property node, which gets or sets a property of the graphical program node specified by the reference to the graphical program node, as claimed. Rather, as noted above, Sojoodi discloses manual creation of the graphical program, as well as manually setting/getting properties of an object via an object node, i.e., a property node. Nowhere does Sojoodi disclose automatically creating a graphical program at all, and specifically fails to disclose using object creation nodes and property node for such purposes.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 308, and so for at least these reasons, claim 308 is allowable.

Claim 309

In addition to the distinctions noted above in regard to claims 285, 305, and 306, the cited art fails to teach "wherein the first graphical program further includes an invoke node; and wherein said automatically creating the new graphical program comprises the invoke node invoking a method on the graphical program node", as recited in claim 309, where the invoke node is used by the graphical program to *automatically* create a graphical program.

In asserting that Sojoodi teaches these features, the Examiner cites col. 6, lines 15-19, which recites:

> Where the object is a software object or component, the object node is executable to either invoke a method of the object, get and/or set one or more properties of the object, or access other capabilities of the object.

Nowhere does the cited text disclose these features of claim 309. While Sojoodi does disclose an object node for invoking a method of the object, Appellant notes that

nowhere does Sojoodi (or McDonald or Yoshida) mention or even hint at using such a node in a graphical program to automatically create a new graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 309, and so claim 309 is allowable.

Claim 310

In addition to the distinctions noted above in regard to claims 285, 305, 306, and 309, the cited art fails to teach "wherein the object creation node outputs a reference to the graphical program node; wherein the invoke node receives as input the reference to the graphical program node; and wherein the invoke node invokes a method on the graphical program node specified by the reference to the graphical program node", as recited in claim 310, where the invoke node and the object creation node are used by the client graphical program to *automatically* create a graphical program.

In asserting that Sojoodi teaches these features, the Examiner cites col. 6, lines 15-19, which recites:

> Where the object is a software object or component, the object node is executable to either invoke a method of the object, get and/or set one or more properties of the object, or access other capabilities of the object.

Nowhere does the cited text disclose these features of claim 310. While Sojoodi does disclose an object node for invoking a method of the object, Appellant notes that nowhere does Sojoodi (or McDonald or Yoshida) mention or even hint at using such a node in a graphical program to automatically create a new graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 310, and so claim 310 is allowable.

Claim 311

In addition to the distinctions noted above in regard to claims 285, 305, 306, and 309, the cited art fails to teach "wherein the invoked method connects the graphical program node to another graphical program node in the new graphical program", as

recited in claim 311, where the invoke node invoking the method is used by the graphical program to *automatically* create a new graphical program.

In asserting that Sojoodi teaches these features, the Examiner cites col. 6, lines 41-47, which recites:

> The graphical program will typically comprise a plurality of nodes, and the method for creating the graphical program comprises arranging on the screen the plurality of nodes, including the object node, and connecting the various nodes to create the graphical program.

Appellant notes that the text just previous to that cited makes it clear that the arranging and connecting the nodes is performed in response to user input, i.e., is performed *manually*, as has been established above. Nowhere does the cited text disclose an invoke node in a first graphical program (which is executable to automatically create a new graphical program) invoking a method to automatically connect the (automatically created) graphical program node to another graphical program node in the new graphical program, as claimed. While Sojoodi does disclose an object node for invoking a method of the object, Appellant notes that nowhere does Sojoodi (or McDonald or Yoshida) mention or even hint at using such a node in a graphical program to automatically create a new graphical program, specifically, to automatically connect nodes in the new graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 311, and so claim 311 is allowable.

Claim 312

In addition to the distinctions noted above in regard to claims 285, 305, 306, 309, and 311, the cited art fails to teach "wherein said connecting the graphical program node to said another graphical program node comprises connecting an input of the graphical program node to an output of said another graphical program node", as recited in claim 312, where the connecting is performed by the graphical program to *automatically* create a new graphical program.

In asserting that Sojoodi teaches these features, the Examiner cites col. 6, lines 41-47, which recites:

> The graphical program will typically comprise a plurality of nodes, and the method for creating the graphical program comprises arranging on the screen the plurality of nodes, including the object node, and connecting the various nodes to create the graphical program.

Appellant notes that the text just previous to that cited makes it clear that the arranging and connecting the nodes is performed in response to user input, i.e., is performed *manually*, as has been established above. Nowhere does the cited text disclose an invoke node in a first graphical program (which is executable to automatically create a new graphical program) invoking a method to automatically connect (automatically created) an input of the graphical program node to an output of another graphical program node in the new graphical program, as claimed. While Sojoodi does disclose an object node for invoking a method of the object, and manually connecting nodes, Appellant notes that nowhere does Sojoodi (or McDonald or Yoshida) mention or even hint at automatically invoking a method for automatically connecting nodes to automatically create a new graphical program, specifically, to automatically connect respective inputs and outputs of nodes in the new graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 312, and so claim 312 is allowable.

Claim 313

In addition to the distinctions noted above in regard to claims 285, 305, 306, and 309, the cited art fails to teach "wherein the invoked method performs one of 1) moving the graphical program node to another location in the new graphical program; and 2) resizing the graphical program node in the new graphical program", as recited in claim 313, where the invoke node invoking the method is used by the graphical program to *automatically* create a new graphical program.

In asserting that Sojoodi teaches these features, the Examiner cites col. 56, lines 38-41, which recites:

> ...wherein the software object is created in a second program development environment, wherein the second program development environment is different than the first graphical program development environment...

Nowhere does the cited text disclose an invoke node in a first graphical program (which is executable to automatically create a new graphical program) invoking a method to move the automatically created graphical program node, or to resize the graphical program node, as claimed. While Sojoodi does disclose an object node for invoking a method of the object, Appellant notes that nowhere does Sojoodi (or McDonald or Yoshida) mention or even hint at using such a node in a graphical program to automatically create a new graphical program, specifically, to automatically move the graphical program node to another location in the new graphical program or to automatically resize the graphical program node in the new graphical program.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 313, and so claim 313 is allowable.

Claim 319, 369

In addition to the distinctions noted above in regard to claims 285 and 305, the cited art fails to teach "wherein the first graphical program includes a graphical program creation node for automatically creating the new graphical program", as recited in claim 319.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 27-30, which recites:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

Clearly, the cited text does not disclose these features of claim 319, since it makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, the first graphical program including a graphical program creation node for automatically instantiating a new graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a graphical program creation node for *automatically* instantiating a new graphical program. Nor does McDonald disclose such a node used for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 319, and so claim 319 is allowable.

Claim 320

In addition to the distinctions noted above in regard to claims 285, 305, and 319, the cited art fails to teach "wherein said creating the first graphical program comprises: displaying the graphical program creation node; and specifying a new graphical program type for the graphical program creation node; wherein said creating the new graphical program comprises creating the new graphical program of the specified new graphical program type", as recited in claim 320.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 27-30, which recites:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

And col. 6, lines 12-15, which reads:

> In the preferred embodiment, the object may be any of various types of software, such as a software object according to the standard principles of object-oriented software, a software component, other types of re-usable software elements, or an application, among others.

Clearly, the cited text does not disclose these features of claim 320, since it makes no mention of automatically creating a graphical program at all, nor the first graphical program including a graphical program creation node for automatically instantiating a new graphical program, and more specifically fails to mention or even hint at creating the first graphical program of a specified new graphical type by displaying the graphical program creation node, and specifying a new graphical program type for the graphical program creation node. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a graphical program creation node for *automatically* instantiating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 320, and so claim 320 is allowable.

Claim 321

In addition to the distinctions noted above in regard to claims 285, 305, 319, and 320, the cited art fails to teach "wherein the graphical program creation node includes a type input; and wherein said specifying a new graphical program type for the graphical program creation node comprises connecting type information to the type input of the graphical program creation node", as recited in claim 321.

In asserting that Sojoodi teaches these features, the Examiner simply states that Sojoodi teaches that the graphical program creation node inherently includes a type input.

Appellant respectfully submits that, as noted above, Sojoodi nowhere discloses automatically creating a graphical program at all, nor the first graphical program including a graphical program creation node for automatically instantiating a new graphical program, nor creating the first graphical program of a specified new graphical type by displaying the graphical program creation node, and specifying a new graphical program type for the graphical program creation node, and more specifically fails to teach or suggest specifying a new graphical program type for the graphical program creation node by connecting type information to a type input of a graphical program creation node. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a graphical program creation node for *automatically* instantiating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 321, and so claim 321 is allowable.

Claim 322

In addition to the distinctions noted above in regard to claims 285, 305, and 319, the cited art fails to teach "wherein said creating the first graphical program comprises:

displaying the graphical program creation node; and specifying a reference to a server program for the graphical program creation node; wherein said creating the new graphical program comprises the server program creating the new graphical program", as recited in claim 322.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 8-10, which recites:

> It is desirable for a program, such as a graphical program, to be able to access functionality provided by an object or server.

Clearly, the cited text does not disclose these features of claim 322, since it makes no mention of automatically creating a graphical program at all, nor the first graphical program including a graphical program creation node for automatically instantiating a new graphical program, and more specifically fails to mention or even hint at displaying the graphical program creation node, specifying a reference to a server program for the graphical program creation node, and the server program creating the new graphical program, as claimed. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a graphical program creation node for *automatically* instantiating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 322, and so claim 322 is allowable.

Claim 323

In addition to the distinctions noted above in regard to claims 285, 305, 319, and 322, the cited art fails to teach "wherein the server program is an application instance of a graphical programming development environment", as recited in claim 323.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 8-10, which recites:

> It is desirable for a program, such as a graphical program, to be able to access functionality provided by an object or server.

Clearly, the cited text does not disclose these features of claim 323, since it makes no mention of automatically creating a graphical program at all, nor the first graphical program including a graphical program creation node for automatically instantiating a new graphical program, displaying the graphical program creation node, specifying a reference to a server program for the graphical program creation node, and the server program creating the new graphical program; and more specifically fails to mention or even hint at the server program (that automatically creates the new graphical program) being an application instance of a graphical programming development environment, as claimed. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a server program *automatically* instantiating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 323, and so claim 323 is allowable.

Claim 324

In addition to the distinctions noted above in regard to claims 285, 305, 319, and 322, the cited art fails to teach "wherein the graphical program creation node includes a server program reference input; and wherein said specifying a reference to a server program for the graphical program creation node comprises connecting information specifying a server program to the server program reference input of the graphical program creation node", as recited in claim 324.

In asserting that Sojoodi teaches these features, the Examiner cites Figure 37: VI Server Refnum Control 274A. However, the associated text describes this item thusly:

> ...the graphical programming environment further comprises VI Server refnum controls 274A. The front panel editor 262 is operable to generate a VI front panel which may include one or more of the VI Server refnum controls 274A. ***The VI Server refnum controls 274A communicate with the VI Server object manager 268 to obtain or provide information regarding class libraries 270 in the system.*** Col. 32, lines 11-18 (*emphasis added*)

Nowhere is the VI Server renum described as being capable of, or used for, such automatic creation of a new graphical program. Appellant notes that Figure 37 is directed to *manually* creating a client graphical program that can programmatically access server capabilities of an application, such as a graphical program application or program, e.g., LabVIEW or a VI. *See, e.g., col. 31, lines 59-63.*

As argued above at length, nowhere does Sojoodi disclose these features of claim 324, since it makes no mention of automatically creating a graphical program at all, nor the first graphical program including a graphical program creation node for automatically instantiating a new graphical program, displaying the graphical program creation node, specifying a reference to a server program for the graphical program creation node, and the server program creating the new graphical program; and more specifically fails to mention or even hint at the graphical program creation node including a server program reference input, and specifying a reference to the server program for the graphical program creation node by connecting information specifying a server program to the server program reference input of the graphical program creation node, as claimed. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a server program *automatically* instantiating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 324, and so claim 324 is allowable.

Claim 325

In addition to the distinctions noted above in regard to claims 285, 305, 319, and 322, the cited art fails to teach "wherein said executing the first graphical program is performed in a first computing system; wherein said server program executes in a second computing system; and wherein the first computing system is connected to the second computing system", as recited in claim 325.

In asserting that McDonald teaches this feature, the Examiner cites step 202 of Figure 2, and col. 5, lines 8-10. Appellant respectfully disagrees.

Step 202 of Figure 2 states, "Select a control which represents input to or output from a graphical program in response to user input". Col. 5, lines 8-10 read:

> It is desirable for a program, such as a graphical program, to be able to access functionality provided by an object or server.

Nowhere do the cited Figure and text mention or even hint at such use of multiple computing systems in the automatic creation of a graphical program as recited in claim 365. More specifically, the cited text fails to disclose a first program executing in a first computing system, and a server program executing in a second computing system connected to the first computer system, where the first program and the server program execute to conjunctively create the new graphical program, as represented in claim 325.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a first program and a server program *automatically* creating a new graphical program in this manner. Nor does Sojoodi disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 325, and so claim 325 is allowable.

Claim 326

In addition to the distinctions noted above in regard to claims 285 and 305, the cited art fails to teach "wherein said creating the first graphical program comprises: displaying a graphical program creation node, wherein the graphical program creation node is operable to automatically create the new graphical program; displaying an object creation node, wherein the object creation node is operable to automatically create at least one graphical program node in the new graphical program; and configuring the object creation node with one or more inputs", as recited in claim 326.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, lines 41-46, and lines 48-53. Appellant respectfully disagrees, and notes that the cited passages actually read (lines 41-53):

> The graphical program will typically comprise a plurality of nodes, and the method for creating the graphical program comprises arranging on the screen the plurality of nodes, including the object node, and connecting

> the various nodes to create the graphical program. In the preferred embodiment, the nodes are connected in a data flow paradigm.
>
> The user then configures the object node to receive information on the object, preferably by the user configuring the object node with a reference to the object, e.g., a pointer, address, or other information which specifies the identity and/or location of the object.

Clearly, the cited text does not disclose these features of claim 326, since it makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, the client graphical program including a graphical program creation node for automatically instantiating a new graphical program, and an object creation node for automatically creating a graphical program node in the new graphical program, as well as configuring the object creation node with one or more inputs, as claimed. Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a server program *automatically* instantiating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Appellant thus respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 326, and so for at least these reasons, claim 326 is allowable.

Claim 327

In addition to the distinctions noted above in regard to claims 285, 305, and 326, the cited art fails to teach "connecting the graphical program creation node to the object creation node; wherein the graphical program creation node outputs a reference to the new graphical program; and wherein said connecting the graphical program creation node to the object creation node comprises connecting the reference to the new graphical program to an input of the object creation node", as recited in claim 327.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, line 66 – col. 6, line 1. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> Creation of the graphical program may also include configuring a user interface to display data input to and/or output from the graphical program.

Clearly, the cited text does not disclose these features of claim 326, since it makes no mention of automatically creating a graphical program at all, nor a graphical program creation node, and more specifically fails to mention or even hint at, connecting the graphical program creation node to an object creation node, where the graphical program creation node outputs a reference to the new graphical program, and where connecting the graphical program creation node to the object creation node includes connecting the reference to the new graphical program to an input of the object creation node, as represented in claim 327. Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a server program *automatically* instantiating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Appellant thus respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 327, and so for at least these reasons, claim 327 is allowable.

Claim 328

In addition to the distinctions noted above in regard to claims 285, 305, and 326, the cited art fails to teach "configuring the graphical program creation node with one or more inputs, wherein said configuring the graphical program creation node with one or more inputs comprises performing one or more of: 1) specifying a new graphical program type for the graphical program creation node; and 2) specifying a server reference for the graphical program creation node", as recited in claim 328.

In asserting that Sojoodi teaches these features, the Examiner cites Figure 37: VI Server Refnum Control 274A. However, the associated text describes this item thusly:

> ...the graphical programming environment further comprises VI Server refnum controls 274A. The front panel editor 262 is operable to generate a VI front panel which may include one or more of the VI Server refnum controls 274A. ***The VI Server refnum controls 274A communicate with the VI Server object manager 268 to obtain or provide information***

regarding class libraries 270 in the system. Col. 32, lines 11-18 (*emphasis added*)

Nowhere is the VI Server renum described as being capable of, or used for, such automatic creation of a new graphical program. Appellant notes that Figure 37 is directed to *manually* creating a client graphical program that can programmatically access server capabilities of an application, such as a graphical program application or program, e.g., LabVIEW or a VI. *See, e.g., col. 31, lines 59-63.*

Nowhere does Sojoodi disclose these features of claim 328, since it makes no mention of automatically creating a graphical program at all, nor the first graphical program including a graphical program creation node for automatically instantiating a new graphical program, and more specifically fails to mention or even hint at configuring the graphical program creation node with one or more inputs, including specifying a new graphical program type for the graphical program creation node, and/or specifying a server reference for the graphical program creation node, as claimed.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose utilizing a graphical program creation node to automatically create a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 328, and so claim 328 is allowable.

Claim 329

In addition to the distinctions noted above in regard to claims 285, 305, 326, and 328, the cited art fails to teach "wherein a server reference is specified for the graphical program creation node; wherein said executing the first graphical program comprises executing program instructions on a first computer; wherein the server reference references a server program running on a second computer; wherein the second computer is connected to the first computer via a network; and wherein said creating the new graphical program in response to said executing the first graphical program comprises the server program creating the new graphical program", as recited in claim 328.

In asserting that Sojoodi teaches these features, the Examiner cites Figure 37: VI Server Refnum Control 274A. However, the associated text describes this item thusly:

> ...the graphical programming environment further comprises VI Server refnum controls 274A. The front panel editor 262 is operable to generate a VI front panel which may include one or more of the VI Server refnum controls 274A. ***The VI Server refnum controls 274A communicate with the VI Server object manager 268 to obtain or provide information regarding class libraries 270 in the system***. Col. 32, lines 11-18 (*emphasis added*)

Nowhere is the VI Server renum described as being capable of, or used for, such automatic creation of a new graphical program. Appellant notes that Figure 37 is directed to *manually* creating a client graphical program that can programmatically access server capabilities of an application, such as a graphical program application or program, e.g., LabVIEW or a VI. *See, e.g., col. 31, lines 59-63.*

Nowhere does Sojoodi disclose these features of claim 329, since it makes no mention of automatically creating a graphical program at all, nor the first graphical program including a graphical program creation node for automatically instantiating a new graphical program, and more specifically fails to mention or even hint at such automatic creation of a new graphical program via execution of the first graphical program on a first computer and execution of a server program (referenced by a server reference specified for the graphical program creation node) on a second computer connected to the first computer via a network, where creating the new graphical program in response to said executing the first graphical program includes the server program creating the new graphical program, as claimed.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose utilizing a graphical program creation node (with a specified server reference) and a server program to automatically create a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 329, and so claim 329 is allowable.

Claim 330

In addition to the distinctions noted above in regard to claims 285, 305, and 326, the cited art fails to teach "wherein said configuring the object creation node with one or more inputs comprises performing one or more of: 1) specifying a node class for the object creation node; 2) specifying a node sub-class for the object creation node; 3) specifying position information to the object creation node; and 4) specifying owner reference information for the object creation node", as recited in claim 330.

In asserting that Sojoodi teaches these features, the Examiner cites col. 13, lines 47-56, and col. 15, lines 32-38. Appellant respectfully disagrees, and notes that the cited passages read:

> The object manager 268 accesses object class/type libraries 270 to acquire information necessary to perform object management operations. Preferably, the object manager 268 communicates with the Windows operating system Registry, or other similar data structures, to obtain information regarding the object class/type libraries 270 in the system. The object control 274, object nodes 266, and object manager 268 will be discussed in more detail below.
>
> Advantageously, the graphical programming environment, and in particular the object control 274, the object nodes 266, the object manager 268, and the object manager run-time library 258, enable a graphical program to instantiate objects of an unlimited number of object classes of object applications, and to remotely invoke properties and methods of the instantiated objects. (col. 13, lines 47-61)

And

> In step 306 the user then configures the object node to receive information on the object, preferably by the user configuring the object node with a reference to the object, e.g., a pointer, address, or other information which specifies the identity and/or location of the object. (col. 15, lines 32-36)

Clearly, the cited text does not disclose these features of claim 330, since it makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, configuring an object creation node with one or more inputs, including: specifying a node class for the object creation node, specifying a node sub-class for the object creation node, specifying position information to the object creation node, and/or specifying owner reference information for the object creation node, as part of automatically creating the new graphical program, as represented in claim 327.

Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 330, and so claim 330 is allowable.

Claim 331

In addition to the distinctions noted above in regard to claims 285 and 305, the cited art fails to teach "wherein the first graphical program includes a plurality of object creation nodes each for automatically creating a graphical program node in the new graphical program, wherein said plurality of object creation nodes includes a first object creation node for creating a first graphical program node in the new graphical program and includes a second object creation node for creating a second graphical program node in the new graphical program; wherein said executing the first graphical program comprises including the first graphical program node and the second graphical program node in the new graphical program; wherein the first graphical program further includes a node operable to connect the first graphical program node to the second graphical program node; and wherein said executing the first graphical program includes connecting the first graphical program node to the second graphical program node", as recited in claim 331.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 5, lines 40-47. Appellant respectfully disagrees, and notes that the cited passages actually read:

> This preferably includes the user arranging on the screen the graphical program, including the object node. The graphical program will typically comprise a plurality of nodes, and the method for creating the graphical program comprises arranging on the screen the plurality of nodes, including the object node, and connecting the various nodes to create the graphical program. In the preferred embodiment, the nodes are connected in a data flow paradigm.

Clearly, the cited text does not disclose these features of claim 331, since it makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, a first graphical program (executable to automatically create a new graphical program) including a plurality of object creation nodes each for automatically creating a graphical program node in the new graphical program. Nor does Sojoodi disclose the plurality of object creation nodes including a first object creation node for creating a first graphical program node in the new graphical program and a second object creation node for creating a second graphical program node in the new graphical program. Nor executing the first graphical program to create the new graphical program, thereby including the first graphical program node and the second graphical program node in the new graphical program, where the first graphical program further includes a node operable to connect the first graphical program node to the second graphical program node, and where executing the first graphical program includes connecting the first graphical program node to the second graphical program node, as represented in claim 331.

Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Appellant thus respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 331, and so for at least these reasons, claim 331 is allowable.

Claim 332

In addition to the distinctions noted above in regard to claims 285 and 305, the cited art fails to teach "wherein the first graphical program includes a graphical program creation node for automatically creating the new graphical program; wherein the first graphical program includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program; and wherein said executing the first graphical program includes: executing the graphical program creation

node, wherein said executing the graphical program creation node causes creation of the new graphical program; and executing the object creation node, wherein said executing the object creation node causes inclusion of the graphical program node in the new graphical program", as recited in claim 332.

As argued at length above, Sojoodi makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, a first graphical program (executable to automatically create a new graphical program) including a graphical program creation node for automatically creating a new graphical program, wherein the first graphical program includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program. Nor does Sojoodi teach or suggest that executing the first graphical program includes: executing the graphical program creation node, thereby causing creation of the new graphical program, and executing the object creation node, thereby causing inclusion of the graphical program node in the new graphical program, as represented in claim 332.

Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Appellant thus respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 332, and so for at least these reasons, claim 332 is allowable.

Claim 353

Claim 353 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

Appellant notes that McDonald is directed to generating a graphical program (or portion) by programmatically selecting *pre-existing code templates* based on user-selected graphical user interface controls and inserting the code templates into a graphical program, which is not at all the same as the present invention as represented in claim 353. The Examiner asserts that McDonald teaches "a first program, when executing,

programmatically creating a new graphical program" [sic], citing col. 3, lines 61-63. Appellant notes that the cited portion of McDonald reads:

> The present invention comprises a computer-implemented system and method for automatically generating graphical code in a graphical programming system.

Appellant submits that there are numerous possible ways to automatically generate graphical code in a graphical programming system, and that the method disclosed by McDonald is not at all the same as that claimed in the present application. Nowhere does McDonald teach or suggest "a first graphical program, wherein the first graphical program includes a graphical program creation node for automatically instantiating a new graphical program, wherein the first graphical program also includes a first object creation node for creating a first node in the new graphical program and includes a second object creation node for creating a second node in the new graphical program; wherein the first graphical program is executable to instantiate the new graphical program, include the first node and the second node in the new graphical program, and connect the first node to the second node, where the first graphical program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes", as claimed.

The Examiner admits that McDonald fails to teach "that the first program is a first graphical program and the creating a new graphical program by creating graphical program objects and by interconnecting these created graphical program objects [sic]", but asserts that this feature is taught by Sojoodi, citing col. 5, lines 40-47. Appellant respectfully disagrees, and notes that the cited passages actually read:

> This preferably includes ***the user*** arranging on the screen the graphical program, including the object node. The graphical program will typically comprise a plurality of nodes, and the method for creating the graphical program comprises arranging on the screen the plurality of nodes, including the object node, and connecting the various nodes to create the graphical program. In the preferred embodiment, the nodes are connected in a data flow paradigm. (*emphasis added*)

Clearly, the cited text does not disclose these features of claim 353, since it makes no mention of automatically creating a graphical program at all, and more specifically

fails to mention or even hint at, a first graphical program (executable to automatically create a new graphical program) including first and second object creation nodes for automatically creating respective graphical program node in the new graphical program. Nor does Sojoodi disclose that the first graphical program is executable to instantiate the new graphical program, include the first node and the second node in the new graphical program, and connect the first node to the second node, as represented in claim 353.

Rather, Sojoodi discloses *manual* creation of a graphical program, as clearly indicated in the cited text. Thus, Sojoodi fails to remedy this admitted deficiency of McDonald.

The Examiner further admits that neither McDonald nor Sojoodi discloses "creating the new program from a program without any user input specifying the nodes and interconnection of the plurality of node [sic]", but then asserts that Yoshida teaches this feature, citing col. 6, lines 39-50, col. 8, lines 33-35 and lines 51-60, and col. 11, lines 28-33. Appellant respectfully disagrees, noting that while Yoshida does describe automatic creation of macros, Yoshida fails to mention or even hint at graphical programs at all, and so cannot possibly provide this missing feature, i.e., Yoshida nowhere teaches or suggests, or even hints at, creating a new graphical program without any user input *specifying the nodes and interconnection of the plurality of nodes*, since Yoshida never discloses graphical program nodes or interconnections among the nodes.

Appellant further respectfully submits that neither McDonald, Sojoodi, nor Yoshida provides a motivation to combine, and so the attempted combination of Sojoodi and Yoshida, and of McDonald, Sojoodi, and Yoshida is improper. For example, nowhere does Sojoodi, or Yoshida suggest the desirability or benefit of automatically creating a graphical program without user input selecting the nodes and connections between the nodes. Appellant notes that the only motivation to combine McDonald and Sojoodi indicated by the Examiner is "to enable the system to create the second graphical program from objects", which Appellant submits is not a proper motivation to combine, lacking both the necessary specificity, and proper support in the references. Appellant notes that the only motivation to combine McDonald, Sojoodi, and Yoshida, indicated by the Examiner is "so that a new program can be automatically generated from an existing program without requiring user to possess a specialized knowledge about computer

language [sic]", which Appellant submits is not a proper motivation to combine. Appellant submits that this statement is much too broad to motivate the particular combination attempted by the Examiner, and is simply a statement of perceived benefit of the alleged combination, and so is an improper motivation to combine. Similarly, nowhere does McDonald indicate the desirability or benefit of automatically creating a graphical program in the manner claimed, e.g., by automatically creating and connecting one or more nodes without user input specifying the nodes or connections.

Thus, for at least these reasons, Appellant submits that McDonald, Sojoodi, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 353, and so claim 353 is allowable.

Claim 354

In addition to the distinctions noted above in regard to claims 353, the cited art fails to teach "wherein the first graphical program also includes a node for obtaining a graphical program node reference; wherein the reference to the existing graphical program is provided to the node for obtaining a graphical program node reference; wherein the node for obtaining a graphical program node reference is configured to obtain a reference to a particular node of the existing graphical program; wherein the first graphical program includes a modify node; wherein the reference to the particular node of the existing graphical program is provided to the modify node; and wherein the modify node is configured to modify the particular node of the existing graphical program", as recited in claim 354.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 54, lines 18-30. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> executing a first graphical program development environment application;
>
> receiving user input to the first graphical program development environment application for creation of the graphical program;
>
> wherein said receiving user input to the first graphical program development environment application comprises:
>
> receiving user input requesting inclusion of a first node in the graphical program;

receiving user input to configure the first node to access capabilities of a pre-existing software object;

Clearly, the cited text does not disclose these features of claim 354, since it makes no mention of executing a first program to automatically create a graphical program at all, and more specifically fails to mention or even hint at, a first graphical program (executable to automatically create a new graphical program) including a node for obtaining a graphical program node reference, where the reference to the existing graphical program is provided to the node for obtaining a graphical program node reference, nor where the node for obtaining a graphical program node reference is configured to obtain a reference to a particular node of the existing graphical program, nor where the first graphical program includes a modify node, where the reference to the particular node of the existing graphical program is provided to the modify node, and where the modify node is configured to modify the particular node of the existing graphical program, as represented in claim 331.

Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose creating a new graphical program in this manner. Nor does McDonald disclose such a process for creating a new graphical program in the manner claimed.

Appellant thus respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 354, and so for at least these reasons, claim 354 is allowable.

Claim 355

Claim 355 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

The Examiner admits that McDonald fails to teach "the creating a graphical program by creating graphical program nodes [sic]", but asserts that this feature is taught by Sojoodi. The Examiner further admits that neither McDonald nor Sojoodi discloses "creating the new program from a program without any user input specifying the nodes

and interconnection of the plurality of node [sic]", but then asserts that Yoshida teaches this feature. Appellant respectfully disagrees, submits that the cited art, taken singly or in combination, fails to teach all the features and limitations of claim 355.

Appellant notes that McDonald is directed to generating a graphical program (or portion) by programmatically selecting *pre-existing code templates* based on user-selected graphical user interface controls and inserting the code templates into a graphical program, which is not at all the same as the present invention as represented in claim 285. The Examiner asserts that McDonald teaches "the creating of a first program, when executing, programmatically creating a new graphical program" [sic], citing col. 3, lines 61-63. However, Appellant notes that the cited portion of McDonald actually reads:

> The present invention comprises a computer-implemented system and method for automatically generating graphical code in a graphical programming system.

Appellant submits that there are numerous possible ways to automatically generate graphical code in a graphical programming system, and that the method disclosed by McDonald is not at all the same as that claimed in the present application.

The graphical code templates selected in McDonald have preconfigured nodes and interconnections, and hence McDonald does not perform, and has no reason to perform, any automatic creation or interconnection of nodes. Nowhere does McDonald teach or suggest "creating a graphical program by creating graphical program nodes", as asserted by the Examiner.

Nor does Sojoodi disclose programmatic creation of a graphical program via execution of a first program, including generation of nodes and interconnection thereof. In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 40-47. Appellant respectfully disagrees, and notes that the cited passages actually read:

> This preferably includes ***the user*** arranging on the screen the graphical program, including the object node. The graphical program will typically comprise a plurality of nodes, and the method for creating the graphical program comprises arranging on the screen the plurality of nodes, including the object node, and connecting the various nodes to create the graphical program. In the preferred embodiment, the nodes are connected in a data flow paradigm.

Clearly, the cited text does not disclose these features of claim 331, since it makes clear that it is the user who adds and interconnects the nodes. In face, Sojoodi makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, a first graphical program (executable to automatically create a new graphical program) including a graphical program creation function for automatically instantiating a new graphical program, and an object creation function for automatically including a node in the new graphical program. Nor does Sojoodi disclose executing the first program, including automatically instantiating the new graphical program, automatically including the node in the new graphical program, and automatically connecting the node to at least one other node in the new graphical program, as represented in claim 355.

Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Thus, Sojoodi fails to teach these features of claim 355.

The Examiner admits that McDonald and Sojoodi fail to teach "creating the new program from a program without any user input specifying the nodes and interconnection of the plurality of node [sic]", but asserts that Yoshida remedies this deficiency. Appellant respectfully disagrees. While Yoshida does describe automatic creation of macros, nowhere does Yoshida disclose a first program that is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes, and, in fact, fails to mention or even hint at graphical programs at all.

Appellant further respectfully submits that neither McDonald, Sojoodi, nor Yoshida provides a motivation to combine, and so the attempted combination of Sojoodi and Yoshida, and of McDonald, Sojoodi, and Yoshida is improper. For example, nowhere does Sojoodi or Yoshida suggest the desirability or benefit of automatically creating a graphical program, i.e., without user input selecting the nodes and connections between the nodes. Appellant notes that the only motivation to combine indicated by the Examiner is "so that a new program can be automatically generated from an existing program without requiring user to possess a specialized knowledge about computer language [sic]", which Appellant submits is not a proper motivation to combine. Appellant submits that this statement is much too broad to motivate the particular

combination attempted by the Examiner, and is simply a statement of perceived benefit of the alleged combination, and so is an improper motivation to combine. Similarly, nowhere does McDonald indicate the desirability or benefit of automatically creating a graphical program in the manner claimed, e.g., by automatically creating and connecting one or more nodes without user input specifying the nodes or connections.

Thus, for at least these reasons, Appellant submits that McDonald, Sojoodi, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 355, and so claim 355 is allowable.

Claim 356

In addition to the distinctions noted above in regard to claim 355, the cited art fails to teach "wherein the first program is a graphical program; wherein the graphical program creation function comprises a graphical program creation node; and wherein the object creation function comprises an object creation node", as recited in claim 356.
the cited art fails to teach "wherein the first program is a first graphical program".

The Examiner admits that McDonald fails to teach "wherein the first program is a first graphical program", but then asserts that Sojoodi remedies this deficiency, citing col. 5, lines 28-32. Appellant respectfully disagrees, noting that the cited text reads:

> The present invention comprises a system and method for creating a graphical program, wherein the graphical program is operable to access capabilities of an object.

Nowhere does the cited portion (or any other portion of Sojoodi) refer to "automatically" creating a graphical program, specifically via execution of a graphical program, as claimed. Thus, Sojoodi does not, and cannot, disclose a first graphical program executable to automatically create a new graphical program, as claimed. Nor does Sojoodi (or McDonald) teach or suggest "wherein the graphical program creation function comprises a graphical program creation node; and wherein the object creation function comprises an object creation node", as recited in claim 356.

Moreover, as noted above, Sojoodi fails to indicate the desirability of such automatic creation of graphical programs. Nor does McDonald suggest the desirability of a graphical program that executes to automatically create a new graphical program.

Thus, Appellant submits that the attempted combination of McDonald and Sojoodi (and Yoshida) is improper.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 356, and so for at least these reasons, claim 356 is allowable.

Claim 357

In addition to the distinctions noted above in regard to claim 355, the cited art fails to teach "wherein the first program is a text-based program", as recited in claim 357.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 15, line 1. Appellant respectfully disagrees, and notes that the cited passage (and surrounding relevant text) actually reads:

> Yet another benefit is the simplicity of programming which makes the development of a graphical program, such as an instrumentation control program, more practical for a larger number of people, i.e., those who might not have the skills, or resources to develop the skills, to develop programs according to more conventional text-based methods.

The cited text describes benefits of the manual graphical programming techniques disclosed by Sojoodi, e.g., dragging and dropping graphical program nodes into a graphical program and interconnecting them, all in response to user input. Clearly, the cited text does not disclose these features of claim 357, since it makes no mention of automatically creating a graphical program at all, and more specifically fails to mention or even hint at, a text-based program that is executable to automatically create a new graphical program). Nor does Sojoodi disclose that the text-based program includes a graphical program creation function for automatically instantiating a new graphical program, and an object creation function for automatically including a node in the new graphical program. Nor does Sojoodi disclose executing the first (text-based) program to automatically instantiat the new graphical program, automatically include the node in the new graphical program, and automatically connecting the node to at least one other node in the new graphical program, where the first program is operable to automatically create the new graphical program without any user input specifying selection of graphical

program nodes and interconnection of graphical program nodes, as represented in claim 357.

Rather, as argued at length above, Sojoodi discloses *manual* creation of a graphical program. Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a text-based program for automatically creating a new graphical program as claimed.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 357, and so for at least these reasons, claim 357 is allowable.

Claim 358

In addition to the distinctions noted above in regard to claims 355 and 357, the cited art fails to teach "wherein the graphical program creation function comprises a method call to create the new graphical program; and wherein the object creation function comprises a method call to create the node", as recited in claim 358.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 31, lines 39 - 41. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> The call by reference node is essentially an invoke node as described above which can invoke or call only a run method on an application.

Clearly, the cited text does not disclose these features of claim 358, since it makes no mention of automatically creating a graphical program at all, and is particularly limited to invoking a "run method" on an application. In other words, the cited reference or invoke node is not capable of creating a new graphical program.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a text-based program for automatically creating a new graphical program as claimed.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 358, and so for at least these reasons, claim 358 is allowable.

Claim 359

In addition to the distinctions noted above in regard to claims 355 and 357, the cited art fails to teach "wherein the text-based program obtains a reference to a software component; wherein the software component enables the text-based program to perform the method call to create the new graphical program; and wherein the software component enables the text-based program to perform the method call to create the node", as recited in claim 359.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 58, lines 42-44. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> ...wherein the pre-existing software object was constructed in response to text-based programming language source code.

Appellant respectfully submits that Sojoodi's pre-existing software object being constructed in response to text-based programming language source code in no way teaches the above claimed features, and notes that Sojoodi nowhere describes this text-based programming language source code as being a text-based program that is executable to automatically instantiate a new graphical program, automatically include the node in the new graphical program, and automatically connect the node to at least one other node in the new graphical program, all without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes. More specifically, Sojoodi fails to teach or suggest the text-based program obtaining a reference to a software component that enables the text-based program to perform the method call to create the new graphical program, and to perform the method call to create the node, as represented in claim 359.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a text-based program for automatically creating a new graphical program as claimed.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 359, and so for at least these reasons, claim 359 is allowable.

Claim 360

In addition to the distinctions noted above in regard to claims 355, 357, and 359 the cited art fails to teach "wherein the software component interfaces with a server program; wherein the server program receives the method call to create the new graphical program; wherein the server program creates the new graphical program; wherein the server program receives the method call to create the node; and wherein the server program creates the node", as recited in claim 360.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 14, lines 41-46. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> For example, the object server 252 may be a program written in the C language for controlling a GPIB instrument. The client VI 250 instantiates objects from the classes exported by the object server 252 and invokes methods and properties of the objects to direct the object server 252 to control the instrument hardware 254.

Clearly, the cited text does not teach these features of claim 360. In fact, as argued at length above, Sojoodi nowhere discloses automatically creating a graphical program at all, and specifically fails to disclose doing so via a text-based program using a software component to make a method call to a server program to automatically create a new graphical program, including creating a node in the new graphical program.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a text-based program and server program for automatically creating a new graphical program as claimed.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 360, and so for at least these reasons, claim 360 is allowable.

Claim 361

In addition to the distinctions noted above in regard to claims 355, 357, and 359 the cited art fails to teach "wherein the software component is an ActiveX component", as recited in claim 360.

In asserting that Sojoodi teaches this feature, the Examiner cites col. 57, line 62. Appellant respectfully disagrees, and notes that the cited passage actually reads:

> ...wherein the software object comprises one of: an ActiveX object.

Appellant respectfully submits that Sojoodi's ActiveX software object is nowhere described as having the functionality recited in claim 361. In fact, as argued at length above, Sojoodi nowhere discloses automatically creating a graphical program at all, and specifically fails to disclose the text-based program obtaining a reference to a software component that enables the text-based program to perform the method call to create the new graphical program, and to perform the method call to create the node.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a text-based program and software component for automatically creating a new graphical program as claimed.

Thus, Appellant respectfully submits that nowhere do McDonald, Sojoodi, and Yoshida, taken singly or in combination, teach or suggest the features and limitations of claim 361, and so for at least these reasons, claim 361 is allowable.

Claim 363

In addition to the distinctions noted above in regard to claim 362 (as presented above in arguments directed to claim 285), the cited art fails to teach "wherein, in response to said CPU executing the first program, the first program is operable to interface with a server program; and wherein the server program is operable to automatically create the new graphical program in response to said interfacing", as recited in claim 363.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 8-10, which recites:

> It is desirable for a program, such as a graphical program, to be able to access functionality provided by an object or server.

Clearly, the cited text does not disclose these features of claim 363, since it makes no mention of automatically creating a graphical program at all, and specifically fails to disclose that the first program is operable to interface with a server program that is

operable to automatically create the new graphical program in response to said interfacing, as represented in claim 363.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a first program being operable to interface with a server program that is operable to automatically create the new graphical program in response to the interfacing. Nor does McDonald disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 363, and so claim 363 is allowable.

Claim 364

In addition to the distinctions noted above in regard to claims 362 (as presented above in arguments directed to claim 285) and 363, the cited art fails to teach "wherein the server program is an application instance of a graphical programming development environment", as recited in claim 364.

In asserting that Sojoodi teaches these features, the Examiner cites col. 5, lines 8-10, which recites:

> It is desirable for a program, such as a graphical program, to be able to access functionality provided by an object or server.

Clearly, the cited text does not disclose these features of claim 364, since it makes no mention of automatically creating a graphical program at all, nor that the first program is operable to interface with a server program that is operable to automatically create the new graphical program in response to said interfacing, and specifically fails to disclose that the server program is an application instance of a graphical programming development environment, as represented in claim 364.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a server program for *automatically* creating a new graphical program in this manner. Nor does McDonald disclose such a server program for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 364, and so claim 364 is allowable.

Claim 365

In addition to the distinctions noted above in regard to claims 362 (as presented above in arguments directed to claim 285) and 363, the cited art fails to teach "wherein said computer system is a first computer system, the system further comprising: a second computer system; wherein the server program executes in the second computer system; and wherein the first computer system is connected to the second computer system via a network", as recited in claim 365.

In asserting that McDonald teaches this feature, the Examiner cites step 202 of Figure 2, and col. 5, lines 8-10. Appellant respectfully disagrees.

Step 202 of Figure 2 states, "Select a control which represents input to or output from a graphical program in response to user input". Col. 5, lines 8-10 read:

> It is desirable for a program, such as a graphical program, to be able to access functionality provided by an object or server.

Nowhere do the cited Figure and text mention or even hint at such use of multiple computer systems in the automatic creation of a graphical program as recited in claim 365. More specifically, the cited text fails to disclose a first program executing in a first computer system, and a server program executing in a second computer system connected to the first computer system via a network, where the first program and the server program execute to conjunctively create the new graphical program, as represented in claim 365.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose a first program and a server program *automatically* creating a new graphical program in this manner. Nor does Sojoodi disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 365, and so claim 365 is allowable.

Claim 366

In addition to the distinctions noted above in regard to claim 362 (as presented above in arguments directed to claim 285), the cited art fails to teach "wherein at least one of the plurality of graphical program nodes comprises a structure node", as recited in claim 366.

In asserting that McDonald teaches this feature, the Examiner cites col. 16, lines 25-27. Appellant respectfully disagrees, noting that col. 16, lines 25-27 read:

> In the preferred embodiment, the locking mechanism is provided only for code encapsulated in a structure node...

Nowhere does the cited text mention or even hint at automatically including a plurality of graphical program nodes in the new graphical program, where at least one of the plurality of graphical program nodes comprises a structure node, as claimed.

Appellant further notes that Yoshida fails to even mention graphical programs at all, and particularly fails to disclose automatically including such a structure node as part of *automatically* creating a new graphical program in this manner. Nor does Sojoodi disclose such a process for automatically creating a new graphical program, as claimed.

Thus, for at least these reasons, Appellant submits that Sojoodi, McDonald, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 366, and so claim 366 is allowable.

Claim 381

Claim 381 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

Appellant notes that McDonald is directed to generating a graphical program (or portion) by programmatically selecting *pre-existing code templates* based on user-selected graphical user interface controls and inserting the code templates into a graphical program, which is not at all the same as the present invention as represented in claim 353. The Examiner asserts that McDonald teaches "a first program, when executing, programmatically creating a new graphical program" [sic], citing col. 3, lines 61-63. Appellant notes that the cited portion of McDonald reads:

> The present invention comprises a computer-implemented system and method for automatically generating graphical code in a graphical programming system.

Appellant submits that there are numerous possible ways to automatically generate graphical code in a graphical programming system, and that the method disclosed by McDonald is not at all the same as that claimed in the present application. Nowhere does McDonald teach or suggest "storing a first graphical program, wherein the first graphical program includes a graphical program creation node for automatically instantiating a new graphical program", nor "wherein the first graphical program also includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program", nor "executing the first graphical program, wherein said executing comprises: automatically instantiating the new graphical program; and automatically including the node in the new graphical program; wherein said first graphical program automatically creating the new graphical program comprises automatically creating the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes; and wherein said automatically including the node in the new graphical program comprises automatically including the node in the new graphical program without any user input specifying inclusion of the node in the new graphical program", as claimed.

The Examiner admits that McDonald fails to teach "that the first program is a first graphical program and the creating a new graphical program by creating graphical program objects and by interconnecting these created graphical program objects [sic]", but asserts that this feature is taught by Sojoodi, citing col. 5, lines 40-47. Appellant respectfully disagrees, and notes that the cited passages actually read:

> This preferably includes ***the user*** arranging on the screen the graphical program, including the object node. The graphical program will typically comprise a plurality of nodes, and the method for creating the graphical program comprises arranging on the screen the plurality of nodes, including the object node, and connecting the various nodes to create the graphical program. In the preferred embodiment, the nodes are connected in a data flow paradigm. (*emphasis added*)

Clearly, the cited text does not disclose these features of claim 381, since it makes no mention of automatically creating a graphical program at all, and more specifically

fails to mention or even hint at, a first graphical program (executable to automatically create a new graphical program) including at least one object creation node for automatically creating at least one graphical program node in the (automatically created) new graphical program.

Nor does Sojoodi disclose that the first graphical program is executable to instantiate the new graphical program, nor that the first graphical program includes at least one object creation node for automatically creating at least one graphical program node in the (automatically created) new graphical program, as represented in claim 381.

Rather, Sojoodi discloses *manual* creation of a graphical program, as clearly indicated in the cited text. Thus, Sojoodi fails to remedy this admitted deficiency of McDonald.

The Examiner further admits that neither McDonald nor Sojoodi discloses "creating the new program from a program without any user input specifying the nodes and interconnection of the plurality of node [sic]", but then asserts that Yoshida teaches this feature, citing col. 6, lines 39-50, col. 8, lines 33-35 and lines 51-60, and col. 11, lines 28-33. Appellant respectfully disagrees, noting that while Yoshida does describe automatic creation of macros, Yoshida fails to mention or even hint at graphical programs at all, and so cannot possibly provide this missing feature, i.e., Yoshida nowhere teaches or suggests, or even hints at, automatically creating the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes, and automatically including the node in the new graphical program without any user input specifying inclusion of the node in the new graphical program, since Yoshida never discloses graphical program nodes or interconnections among the nodes at all.

Appellant further respectfully submits that neither McDonald, Sojoodi, nor Yoshida provides a motivation to combine, and so the attempted combination of Sojoodi and Yoshida, and of McDonald, Sojoodi, and Yoshida is improper. For example, nowhere does Sojoodi, or Yoshida suggest the desirability or benefit of automatically creating a graphical program without user input selecting the nodes and connections between the nodes. Appellant notes that the only motivation to combine McDonald and Sojoodi indicated by the Examiner is "to enable the system to create the second graphical

program from objects", which Appellant submits is not a proper motivation to combine, lacking both the necessary specificity, and proper support in the references. Appellant notes that the only motivation to combine McDonald, Sojoodi, and Yoshida indicated by the Examiner is "so that a new program can be automatically generated from an existing program without requiring user to possess a specialized knowledge about computer language [sic]", which Appellant submits is not a proper motivation to combine. Appellant submits that this statement is much too broad to motivate the particular combination attempted by the Examiner, and is simply a statement of perceived benefit of the alleged combination, and so is an improper motivation to combine. Similarly, nowhere does McDonald indicate the desirability or benefit of automatically creating a graphical program in the manner claimed, e.g., by automatically instantiating the graphical program, including creating one or more nodes without user input specifying the nodes or connections.

Thus, for at least these reasons, Appellant submits that McDonald, Sojoodi, and Yoshida, taken singly or in combination, fail to teach or suggest the subject matter of claim 381, and so claim 381 is allowable.

Regarding the Examiner's comments regarding Appellant's previous arguments directed to claim 370, specifically, Appellant's statement that "Nowhere does the cited portion refer to a server program operable to receive client program calls to automatically create a graphical program based on received information and operable to perform the respective operations, i.e., to automatically create the graphical program", the Examiner only quoted a portion of Appellant's statement in the Final Action, specifically: "nowhere does the cited portion refer to a server program operable to receive client program calls...", then made the assertion that "Sojoodi's teaching 'A server is software application which makes available its classes to be accessed from another application, referred to as a client' means client application (client program) accesses classes (performs API calls to) of server application (server program). In other words, server program receives client program calls. [sic]".

Appellant respectfully submits that the Examiner has improperly characterized Appellant's statement (by omission of crucial details), and further submits that a server

program simply receiving client program calls is well known in the art, and that such well-known server functionality in no way teaches or suggests the server functionality recited in claim 370, specifically, a server program operable to receive the client program calls to automatically create the graphical program based on the received information, wherein, in automatically creating the graphical program, the server program is executable to: automatically create a plurality of graphical program nodes in the new graphical program; and automatically interconnect the plurality of graphical program nodes in the new graphical program, wherein the interconnected plurality of graphical program nodes comprise at least a portion of the new graphical program; wherein the server program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes, as claimed.

Thus, Appellant respectfully submits that the Examiner's arguments do not support the rejection of this claim, nor of any of the other claims presented herein.

VIII. CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 285-381 was erroneous, and reversal of his decision is respectfully requested.

The Commissioner is authorized to charge the appeal brief fee of $500.00 and any other fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5150-37301/JCH. This Appeal Brief is submitted with a return receipt postcard.

Respectfully submitted,

Jeffrey C. Hood
Reg. No. 35,198
ATTORNEY FOR APPLICANT(S)

Meyertons Hood Kivlin Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800

Date: 2/21/2006 JCH/MSW

IX. CLAIMS APPENDIX

The following lists claims 1-381, incorporating entered amendments, including claims 285-381, as on appeal.

1-241. (Cancelled)

242-254. (Withdrawn)

255-262. (Cancelled)

263. (Withdrawn)

264-282. (Cancelled)

283. (Withdrawn)

284. (Cancelled)

285. (Previously Presented) A computer-implemented method for programmatically creating a graphical program, comprising:

storing a first program, wherein the first program is executable to automatically create a new graphical program based on received information;

executing the first program, wherein said executing comprises automatically creating the new graphical program based on the received information, wherein said programmatically creating the new graphical program comprises:

automatically creating a plurality of graphical program nodes in the new graphical program; and

automatically interconnecting the plurality of graphical program nodes in the new graphical program;

wherein the interconnected plurality of graphical program nodes comprise at least a portion of the new graphical program; and

wherein said automatically creating the new graphical program creates the new graphical program without any user input specifying the plurality of graphical program nodes or the interconnection of the plurality of graphical program nodes during said creating.

286. (Previously Presented) The method of claim 285, wherein the new graphical program comprises a diagram portion comprising the plurality of interconnected nodes and a user interface portion; and

wherein said automatically creating the new graphical program comprises creating the diagram portion and the user interface portion.

287. (Previously Presented) The method of claim 285, wherein the new graphical program comprises a data flow diagram.

288. (Previously Presented) The method of claim 285, wherein the new graphical program is a virtual instrument.

289. (Previously Presented) The method of claim 285, wherein said executing the first program occurs in a first computing environment;

wherein said first computing environment is connected to a second computing environment;

wherein said executing the first program comprises sending information from the first computing environment to the second computing environment; and

wherein the new graphical program is created in the second computing environment.

290. (Previously Presented) The method of claim 285,

wherein the first program specifies creation of the new graphical program.

291. (Previously Presented) The method of claim 285, further comprising:

receiving information from a user;

wherein said automatically creating comprises automatically creating the new graphical program at least partially based on the information received from the user.

292. (Previously Presented) The method of claim 285,

wherein the first program comprises a client program that calls an application programming interface (API) to automatically create the new graphical program; and

wherein said executing comprises the first program calling the API to perform said creating and said interconnecting.

293. (Previously Presented) The method of claim 285,

wherein the first program comprises a server program; and

wherein said executing comprises the first program receiving calls from a client program.

294. (Previously Presented) The method of claim 285, wherein the new graphical program comprises a block diagram, and wherein the plurality of graphical program nodes comprise at least one function node placed in the block diagram.

295. (Previously Presented) The method of claim 285, wherein the new graphical program includes a block diagram, and wherein the plurality of graphical program nodes comprise at least one programmatic structure placed in the block diagram.

296. (Previously Presented) The method of claim 285, wherein the plurality of graphical program nodes comprise at least one graphical loop structure.

297. (Previously Presented) The method of claim 285, wherein the plurality of graphical program nodes comprise at least one graphical case structure.

298. (Previously Presented) The method of claim 285, wherein said interconnecting the plurality of graphical program nodes comprises displaying a connection between an input of a first graphical program node and an output of a second graphical program node.

299. (Previously Presented) The method of claim 285, wherein said automatically creating the new graphical program comprises:

automatically creating one or more user interface nodes, wherein the one or more user interface nodes perform one or more of providing input to or displaying output from the new graphical program.

300. (Previously Presented) The method of claim 285, wherein the new graphical program includes a user interface panel, and wherein plurality of graphical program nodes comprise at least one user interface node placed in the user interface panel.

301. (Previously Presented) The method of claim 300, wherein the user interface node comprises at least one of:

a user interface input node placed in the user interface panel for providing user input to the new graphical program; and/or

a user interface output node placed in the user interface panel for viewing output of the new graphical program.

302. (Previously Presented) The method of claim 285, wherein said automatically creating the new graphical program further comprises:

obtaining a reference to a graphical program node, wherein the reference is used to manipulate the graphical program node.

303. (Previously Presented) The method of claim 285, wherein said automatically creating the new graphical program further comprises:

performing at least one of getting or setting a property of a graphical program node.

304. (Previously Presented) The method of claim 285, wherein said automatically creating the new graphical program further comprises:

invoking a method on a graphical program node.

305. (Previously Presented) The method of claim 285, wherein the first program is a first graphical program.

306. (Previously Presented) The method of claim 305, wherein the first graphical program includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program; and

wherein said automatically creating the new graphical program comprises including the at least one graphical program node in the new graphical program.

307. (Previously Presented) The method of claim 306, wherein the first graphical program further includes a property node; and

wherein said automatically creating the new graphical program comprises the property node getting or setting a property of the graphical program node.

308. (Previously Presented) The method of claim 307, wherein the object creation node outputs a reference to the graphical program node;

wherein the property node receives as input the reference to the graphical program node; and

wherein the property node gets or sets a property of the graphical program node specified by the reference to the graphical program node.

309. (Previously Presented) The method of claim 306, wherein the first graphical program further includes an invoke node; and

wherein said automatically creating the new graphical program comprises the invoke node invoking a method on the graphical program node.

310. (Previously Presented) The method of claim 309, wherein the object creation node outputs a reference to the graphical program node;

wherein the invoke node receives as input the reference to the graphical program node; and

wherein the invoke node invokes a method on the graphical program node specified by the reference to the graphical program node.

311. (Previously Presented) The method of claim 309, wherein the invoked method connects the graphical program node to another graphical program node in the new graphical program.

312. (Previously Presented) The method of claim 311, wherein said connecting the graphical program node to said another graphical program node comprises connecting an input of the graphical program node to an output of said another graphical program node.

313. (Previously Presented) The method of claim 309, wherein the invoked method performs one of 1) moving the graphical program node to another location in the new graphical program; and 2) resizing the graphical program node in the new graphical program.

314. (Previously Presented) The method of claim 306, wherein said creating the first graphical program comprises:

displaying the object creation node; and

specifying a graphical program node class for the object creation node;

wherein the at least one graphical program node included in the new graphical program is of the graphical program node class.

315. (Previously Presented) The method of claim 314, wherein said creating the first graphical program further comprises:

specifying a graphical program node sub-class for the object creation node;

wherein the graphical program node included in the new graphical program is of the node sub-class.

316. (Previously Presented) The method of claim 306, wherein said creating the first graphical program comprises:

displaying the object creation node; and

specifying position information to the object creation node;

wherein the at least one graphical program node included in the new graphical program is positioned in the new graphical program at a location based on the position information.

317. (Previously Presented) The method of claim 306, wherein said creating the first graphical program comprises:

displaying the object creation node; and

specifying owner reference information for the object creation node, wherein the owner reference information designates an owner entity;

wherein the at least one graphical program node is included in the new graphical program as a member of the owner entity.

318. (Previously Presented) The method of claim 317, wherein the owner entity is an entity from the group consisting of: 1) the new graphical program; and 2) another graphical program node of the new graphical program.

319. (Previously Presented) The method of claim 305, wherein the first graphical program includes a graphical program creation node for automatically creating the new graphical program.

320. (Previously Presented) The method of claim 319, wherein said creating the first graphical program comprises:

displaying the graphical program creation node; and

specifying a new graphical program type for the graphical program creation node;

wherein said creating the new graphical program comprises creating the new graphical program of the specified new graphical program type.

321. (Previously Presented) The method of claim 320, wherein the graphical program creation node includes a type input; and

wherein said specifying a new graphical program type for the graphical program creation node comprises connecting type information to the type input of the graphical program creation node.

322. (Previously Presented) The method of claim 319, wherein said creating the first graphical program comprises:

displaying the graphical program creation node; and

specifying a reference to a server program for the graphical program creation node;

wherein said creating the new graphical program comprises the server program creating the new graphical program.

323. (Previously Presented) The method of claim 322, wherein the server program is an application instance of a graphical programming development environment.

324. (Previously Presented) The method of claim 322, wherein the graphical program creation node includes a server program reference input; and

wherein said specifying a reference to a server program for the graphical program creation node comprises connecting information specifying a server program to the server program reference input of the graphical program creation node.

325. (Previously Presented) The method of claim 322, wherein said executing the first graphical program is performed in a first computing system;

wherein said server program executes in a second computing system; and

wherein the first computing system is connected to the second computing system.

326. (Previously Presented) The method of claim 305, wherein said creating the first graphical program comprises:

displaying a graphical program creation node, wherein the graphical program creation node is operable to automatically create the new graphical program;

displaying an object creation node, wherein the object creation node is operable to automatically create at least one graphical program node in the new graphical program; and

configuring the object creation node with one or more inputs.

327. (Previously Presented) The method of claim 326, further comprising:

connecting the graphical program creation node to the object creation node;

wherein the graphical program creation node outputs a reference to the new graphical program; and

wherein said connecting the graphical program creation node to the object creation node comprises connecting the reference to the new graphical program to an input of the object creation node.

328. (Previously Presented) The method of claim 326, further comprising configuring the graphical program creation node with one or more inputs, wherein said configuring the graphical program creation node with one or more inputs comprises performing one or more of:

1) specifying a new graphical program type for the graphical program creation node; and 2) specifying a server reference for the graphical program creation node.

329. (Previously Presented) The method of claim 328, wherein a server reference is specified for the graphical program creation node;

wherein said executing the first graphical program comprises executing program instructions on a first computer;

wherein the server reference references a server program running on a second computer;

wherein the second computer is connected to the first computer via a network; and

wherein said creating the new graphical program in response to said executing the first graphical program comprises the server program creating the new graphical program.

330. (Previously Presented) The method of claim 326, wherein said configuring the object creation node with one or more inputs comprises performing one or more of:

1) specifying a node class for the object creation node; 2) specifying a node sub-class for the object creation node; 3) specifying position information to the object creation node; and 4) specifying owner reference information for the object creation node.

331. (Previously Presented) The method of claim 305, wherein the first graphical program includes a plurality of object creation nodes each for automatically creating a graphical program node in the new graphical program, wherein said plurality of object creation nodes includes a first object creation node for creating a first graphical program node in the new graphical program and includes a second object creation node for creating a second graphical program node in the new graphical program;

wherein said executing the first graphical program comprises including the first graphical program node and the second graphical program node in the new graphical program;

wherein the first graphical program further includes a node operable to connect the first graphical program node to the second graphical program node; and

wherein said executing the first graphical program includes connecting the first graphical program node to the second graphical program node.

332. (Previously Presented) The method of claim 305,

wherein the first graphical program includes a graphical program creation node for automatically creating the new graphical program;

wherein the first graphical program includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program; and

wherein said executing the first graphical program includes:

executing the graphical program creation node, wherein said executing the graphical program creation node causes creation of the new graphical program; and

executing the object creation node, wherein said executing the object creation node causes inclusion of the graphical program node in the new graphical program.

333. (Previously Presented) A memory medium, comprising:

a first program, wherein the first program is operable to automatically create a new graphical program based on received information;

wherein the first program comprises program instructions that are operable to:

create a plurality of graphical program nodes in the new graphical program; and

interconnect the plurality of graphical program nodes in the new graphical program;

wherein the interconnected plurality of graphical program nodes comprise at least a portion of the new graphical program; and

wherein the first program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes.

334. (Previously Presented) The memory medium of claim 333, wherein the new graphical program comprises a diagram portion comprising the plurality of interconnected nodes and a user interface portion; and

wherein the first program is operable to automatically create the diagram portion and the user interface portion.

335. (Previously Presented) The memory medium of claim 333, wherein the new graphical program comprises a data flow diagram.

336. (Previously Presented) The memory medium of claim 333, wherein the new graphical program is a virtual instrument.

337. (Previously Presented) The memory medium of claim 333, wherein the first program is operable to execute in a first computing environment;

wherein said first computing environment is connected to a second computing environment;

wherein during execution the first program is operable to send information from the first computing environment to the second computing environment; and

wherein the new graphical program is created in the second computing environment.

338. (Previously Presented) The memory medium of claim 333

wherein the first program specifies creation of the new graphical program.

339. (Previously Presented) The memory medium of claim 333,

wherein the first program is operable to receive information from a user and automatically create the new graphical program at least partially based on the information received from the user.

340. (Previously Presented) The memory medium of claim 333,

wherein the first program comprises a client program that calls an application programming interface (API) to automatically create the new graphical program; and

wherein during execution the first program is operable to call the API to perform said creating and said interconnecting.

341. (Previously Presented) The memory medium of claim 333,

wherein the first program comprises a server program; and

wherein during execution the first program is operable to receive calls from a client program.

342. (Previously Presented) The memory medium of claim 333, wherein the new graphical program comprises a block diagram, and wherein the plurality of graphical program nodes comprise at least one function node placed in the block diagram.

343. (Previously Presented) The memory medium of claim 333, wherein the new graphical program includes a block diagram, and wherein the plurality of graphical program nodes comprise at least one programmatic structure placed in the block diagram.

344. (Previously Presented) The memory medium of claim 333, wherein the plurality of graphical program nodes comprise at least one graphical loop structure.

345. (Previously Presented) The memory medium of claim 333, wherein the plurality of graphical program nodes comprise at least one graphical case structure.

346. (Previously Presented) The memory medium of claim 333, wherein, in interconnecting the plurality of graphical program nodes, the program instructions are operable to display a connection between an input of a first graphical program node and an output of a second graphical program node.

347. (Previously Presented) The memory medium of claim 333, wherein the program instructions are further operable to:

create one or more user interface nodes, wherein the one or more user interface nodes perform one or more of providing input to or displaying output from the new graphical program.

348. (Previously Presented) The memory medium of claim 333, wherein the new graphical program includes a user interface panel, and wherein plurality of graphical program nodes comprise at least one user interface node placed in the user interface panel.

349. (Previously Presented) The memory medium of claim 348, wherein the user interface node comprises at least one of:

a user interface input node placed in the user interface panel for providing user input to the new graphical program; and/or

a user interface output node placed in the user interface panel for viewing output of the new graphical program.

350. (Previously Presented) The memory medium of claim 333, wherein the program instructions are further operable to:

obtain a reference to a graphical program node, wherein the reference is used to manipulate the graphical program node.

351. (Previously Presented) The memory medium of claim 333, wherein the program instructions are further operable to:

perform at least one of getting or setting a property of a graphical program node.

352. (Previously Presented) The memory medium of claim 333, wherein the program instructions are further operable to:

invoke a method on a graphical program node.

353. (Previously Presented) A memory medium, comprising:

a first graphical program, wherein the first graphical program includes a graphical program creation node for automatically instantiating a new graphical program, wherein the first graphical program also includes a first object creation node for creating a first node in the new graphical program and includes a second object creation node for creating a second node in the new graphical program;

wherein the first graphical program is executable to:

instantiate the new graphical program;

include the first node and the second node in the new graphical program; and

connect the first node to the second node; and

wherein the first graphical program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes.

354. (Previously Presented) The memory medium of claim 353, wherein the first graphical program also includes a node for obtaining a graphical program node reference;

wherein the reference to the existing graphical program is provided to the node for obtaining a graphical program node reference;

wherein the node for obtaining a graphical program node reference is configured to obtain a reference to a particular node of the existing graphical program;

wherein the first graphical program includes a modify node;

wherein the reference to the particular node of the existing graphical program is provided to the modify node; and

wherein the modify node is configured to modify the particular node of the existing graphical program.

355. (Previously Presented) A computer-implemented method for automatically creating a graphical program, comprising:

storing a first program, wherein the first program includes a graphical program creation function for automatically instantiating a new graphical program, wherein the first program also includes an object creation function for automatically including a node in the new graphical program; and

executing the first program, wherein said executing comprises:

automatically instantiating the new graphical program;

automatically including the node in the new graphical program; and

automatically connecting the node to at least one other node in the new graphical program;

wherein the first program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes.

356. (Previously Presented) The method of claim 355, wherein the first program is a graphical program;

wherein the graphical program creation function comprises a graphical program creation node; and

wherein the object creation function comprises an object creation node.

357. (Previously Presented) The method of claim 355, wherein the first program is a text-based program.

358. (Previously Presented) The method of claim 357, wherein the graphical program creation function comprises a method call to create the new graphical program; and

wherein the object creation function comprises a method call to create the node.

359. (Previously Presented) The method of claim 357, wherein the text-based program obtains a reference to a software component;

wherein the software component enables the text-based program to perform the method call to create the new graphical program; and

wherein the software component enables the text-based program to perform the method call to create the node.

360. (Previously Presented) The method of claim 359, wherein the software component interfaces with a server program;

wherein the server program receives the method call to create the new graphical program;

wherein the server program creates the new graphical program;

wherein the server program receives the method call to create the node; and

wherein the server program creates the node.

361. (Previously Presented) The method of claim 359, wherein the software component is an ActiveX component.

362. (Previously Presented) A system for automatically creating a graphical program, comprising:

a computer system including a CPU and memory;

wherein the memory stores a first program, wherein the first program specifies creation of a new graphical program, wherein the first program is executable to automatically create the new graphical program;

wherein the CPU is operable to execute the first program to automatically create the new graphical program, wherein, in executing the first program, the CPU is operable to:

create a plurality of graphical program nodes in the new graphical program; and

interconnect the plurality of graphical program nodes in the new graphical program;

wherein the interconnected plurality of graphical program nodes comprise at least a portion of the new graphical program; and

wherein the first program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes.

363. (Previously Presented) The system of claim 362, wherein, in response to said CPU executing the first program, the first program is operable to interface with a server program; and

wherein the server program is operable to automatically create the new graphical program in response to said interfacing.

364. (Previously Presented) The system of claim 363, wherein the server program is an application instance of a graphical programming development environment.

365. (Previously Presented) The system of claim 363, wherein said computer system is a first computer system, the system further comprising:

a second computer system;

wherein the server program executes in the second computer system; and

wherein the first computer system is connected to the second computer system via a network.

366. (Previously Presented) The system of claim 362,

wherein at least one of the plurality of graphical program nodes comprises a structure node.

367. (Previously Presented) The system of claim 362, wherein the first program is a first graphical program.

368. (Previously Presented) The system of claim 367, wherein the first graphical program includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program; and

wherein said creating the new graphical program comprises including the at least one graphical program node in the new graphical program in response to said executing the first graphical program.

369. (Previously Presented) The system of claim 367, wherein the first graphical program includes a graphical program creation node for automatically instantiating the new graphical program.

370. (Previously Presented) A system for automatically creating a graphical program, comprising:

a computer system including a CPU and memory;

a client program executing in the computer system, wherein the client program performs API calls to automatically create a graphical program based on received information; and

a server program operable to receive the client program calls to automatically create the graphical program based on the received information, wherein, in automatically creating the graphical program, the server program is executable to:

automatically create a plurality of graphical program nodes in the new graphical program; and

automatically interconnect the plurality of graphical program nodes in the new graphical program, wherein the interconnected plurality of graphical program nodes comprise at least a portion of the new graphical program;

wherein the server program is operable to automatically create the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes.

371. (Previously Presented) The system of claim 370, wherein the server program executes on another computer system, and wherein said another computer system is connected to said computer system via a network.

372. (Previously Presented) The system of claim 370,

wherein the client program performs said calls to automatically create a graphical program by obtaining a reference to a software component and invoking methods of the software component; and

wherein the software component is operable to perform the operations of automatically creating the graphical program.

373. (Previously Presented) The system of claim 370,

wherein the client program performs said calls to automatically create a graphical program by obtaining a reference to a software component and invoking methods of the software component; and

wherein the software component relays the client program calls to the server program.

374. (Previously Presented) The system of claim 370, wherein the server program is a graphical programming environment application.

375. (Previously Presented) The system of claim 370, wherein the client program is a client graphical program;

wherein the client graphical program includes a graphical program creation node for automatically instantiating a new graphical program;

wherein the client graphical program also includes an object creation node for automatically creating a graphical program node in the new graphical program; and

wherein said API calls to automatically create a graphical program comprise calls resulting from executing the graphical program creation node and the object creation node.

376. (Previously Presented) The system of claim 375, wherein the client graphical program further includes a property node for getting or setting a property of the graphical program node.

377. (Previously Presented) The system of claim 375, wherein the client graphical program further includes an invoke node for invoking a method on the graphical program node.

378. (Previously Presented) The system of claim 377, wherein the object creation node is a first object creation node for automatically creating a first graphical program node in the new graphical program;

wherein the graphical program also includes a second object creation node for automatically creating a second graphical program node in the new graphical program; and

wherein the invoked method connects the first graphical program node to the second graphical program node.

379. (Previously Presented) A memory medium comprising a client program for automatically creating a new graphical program, wherein the client program comprises:

a means for automatically instantiating the new graphical program;

a means for automatically creating a node in the new graphical program;

a means for getting or setting properties of the new graphical program or the node; and

a means for automatically invoking methods on the new graphical program or the node;

wherein said means for automatically instantiating the new graphical program are operable to automatically instantiate the new graphical program without any user input specifying instantiation of the new graphical program;

wherein said means for automatically creating the node in the new graphical program are operable to automatically create the node in the new graphical program without any user input specifying creation of the node in the graphical program; and

wherein said means for automatically invoking methods on the new graphical program or the node are operable to automatically invoke methods on the new graphical program or the node without any user input specifying invocation of methods on the new graphical program or the node.

380. (Previously Presented) The memory medium of claim 379, wherein the client program is a graphical program;

wherein said means for automatically instantiating the new graphical program comprises a graphical program creation node;

wherein said means for automatically creating a node in the new graphical program comprises an object creation node;

wherein said means for getting or setting properties of the new graphical program or the node comprises a property node; and

wherein said means for automatically invoking methods on the new graphical program or the node comprises an invoke node.

381. (Previously Presented) A computer-implemented method for automatically creating a graphical program, comprising:

storing a first graphical program, wherein the first graphical program includes a graphical program creation node for automatically instantiating a new graphical program, wherein the first graphical program also includes at least one object creation node for automatically creating at least one graphical program node in the new graphical program;

executing the first graphical program, wherein said executing comprises:

automatically instantiating the new graphical program; and

automatically including the node in the new graphical program;

wherein said first graphical program automatically creating the new graphical program comprises automatically creating the new graphical program without any user input specifying selection of graphical program nodes and interconnection of graphical program nodes; and

wherein said automatically including the node in the new graphical program comprises automatically including the node in the new graphical program without any user input specifying inclusion of the node in the new graphical program.

X. EVIDENCE APPENDIX

No evidence submitted under 37 CFR §§ 1.130, 1.131 or 1.132 or otherwise entered by the Examiner is relied upon in this appeal.

XI. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.